

المؤسسة الجامعية للدراسات والنشر والتوزيع مجد

جاك ريفيير

البرمجة بلغة المؤول
(الاسمبلر)

ترجمة د. عبد الحسن الحسيني

البرمجة بلغة المؤول
(الاسمبلر)

الطبعة الأولى

1410 هـ ـ 1990 م

المؤسسة الجامعية للدراسات والنشر والتوزيع

بيروت ـ الحمراء ـ شارع اميل اده ـ بناية سلام
هاتف: ٨٠٢٤٢٨ ـ ٨٠٢٤٠٧ ـ ٨٠٢٢٩٦
بيروت ـ المصيطبة ـ بناية طاهر هاتف ٣٠١٠٣٠ ـ ٣١١٣١٠
ص . ب ٦٣١١ / ١١٣ تلكس 20665LE ـ ٢٠٦٨٠ لبنان

سلسلة بإشراف
د. عبد الحسن الحسيني

جاك ريفيير

البرمجة بلغة المؤول (الاسمبلر)

ترجمة د. عبد الحسن الحسيني

المؤسسة الجامعية للدراسات والنشر والتوزيع

هذا الكتاب ترجمة :

LA PROGRAMMATION
EN ASSEMBLEUR

Par

Jacques RIVIERE

تقديم

تعتبر لغة أسيمبلر (المؤوّل) من اللغات الفعالة وذات الإمكانيات الكبيرة نظراً لأنها تسمح للمبرمج باستعمال جميع إمكانيات ومقدرات وموارد الحاسب، كما تسمح له بالدخول إلى «قلب» الآلة والعمل بالمراصف الداخلية للحاسب، مما يضفي على البرنامج المكتوب بهذه اللغة فعالية كبيرة خصوصاً فيما يتعلّق بالدقة والسرعة والعمل في الوقت الفعلي (real time) المستعمل كثيراً لإدارة العمليات الصناعية.

هذا الكتاب يُعالج لغة أسمبلر الخاصة بعائلة الحاسبات IBM 360/370 التي شهدت إنتشاراً واسعاً في حقل المعلوماتية وأحدثت ثورة في صناعة الحاسبات في السنوات الأخيرة وبقيت تركيبة وهيكلية هذه الآلات مُستعملة وصالحة في وقتنا هذا وجرى إستعمالها والإفادة منها حتى في صناعة المعالج الصغري وتصميم الميكروحاسبات.

وبالنسبة للبرمجة بلغة المؤوّل، فإن تقنية هذه البرمجة لا تختلف أبداً من آلة إلى أخرى، صغيرة كانت أم كبيرة، معالجاً صغرياً أو نظاماً كبيراً. أما الفرق الوحيد فيكمن في كون كود ـ الآلة يختلف من الة إلى أخرى، أما طريقة العمل والمعالجة وإستعمال المراصف والذاكرة فلا تختلف إلا في عدد المراصف المبلوغة من المُبرمج، وبالتالى فإن التصرّف على أي مؤوّل يبقى صالحاً بالنسبة لمُعالج آخر بمؤوّل آخر.

وهنا يجب الإشارة إلى أن مؤول IBM/370 يتألف من أكبر عدد ممكن من التعليمات، وعدد مراصف الحاسب يعادل 16 للمعطيات و16 للعناوين ويستعمل عدداً كبيراً من طرق العنونة، يصلح قسم منها لعنونة المعلومات عند إستعمال المعالج الصغري.

المترجم

تمهيد

لماذا كتاب جديد يختص بلغة المؤوّل (Assembler) ؟ وما هو المؤوّل ؟ هل تعرفون مُبرمجين يعملون بلغة المؤول حتى الآن ، بينما تقدِّم اللغات المتطورة إمكانيات وتسهيلات جديدة ؟

كثيراً ما نسمع جميع هذه الأسئلة إضافة إلى أخرى مدهشة ، ولن نحاول هنا في هذا التمهيد أن نجاوب عنها ، السؤال بعد الآخر ، ولكن سنحاول توضيح هدفنا من هذا الكتاب .

وُضع هذا الكتاب بسبب ثلاث ملاحظات :

- إن إتقان لغة المؤول هو الطريقة الأفضل لفهم طريقة عمل الحاسب .
- بواسطة إتقان لغة المؤول ، مهما يكن ، سنستطيع التفكير بسهولة أكثر وإدراك ماذا يحدث عندما نعمل بلغة أكثر تطوراً ، والبحث عن الأخطاء سيكون أكثر سهولة .
- عند نزول الميكروبروسسور إلى الأسواق ، أليس من الأفضل إتقان هذه اللغة الموجودة على هذه الآلات الصغيرة ؟ مع الإشارة إلى أن المؤول يبقى الوسيلة الفضلى لإنشاء وخلق المناهج الجديدة .

هكذا فلكتابنا هذا هدف تربوي . وهو ليس عبارة عن كتاب مساعد ومرجع في المعنى الذي نفهمه من المرجع المساعد الخاص بالمنتج ، ولكنه عبارة عن مساعدٍ كافٍ وكامل لفهم عمليات الإنشاء والبرمجة المهمة .

وهو موجّهٌ إلى أولئك الراغبين بفهم طريقة عمل الآلات التي يستعملونها . ولقد حاولنا الإجابة عن المسائل التي ستواجهنا ، وبشكل خاص لدى الطلاب الذين يرغبون بمعرفة لغة المؤول بعد معرفتهم بإحدى اللغات المتطورة . وهذا هو دور الفصل الأوّل من الكتاب الذي يحتوي على عرض لتركيبة وطريقة عمل الحاسب ، وهذا العرض جرى من خلال تفكير بسيط يتعلق بآلة ذات إستعمال كبير : الحاسب الجيبي . ولأجل هؤلاء

أيضاً قمنا بعرض مشاكل العنونة ، التقطيع ، تنقيح الأربطة (link editor) ، الشحن (loading) ، والإنقطاعات عند الإدخال والإخراج (I/O interruption) .

وهـو موجـه أيضاً الى كـل مـن يـرغـب بالعمل بلغـة المـؤوّل ، إمـا علـى الآلة المعتمدة كمرجع وهي الحاسب IBM 370 ، أو على الحاسب الشخصي الميكروكومبيوتر . وهنا نؤكد بأن جميع لغات التأويل هي متشابهة بشكل نستطيع معه بعد معرفة مؤوّل معيّـن أن نتكيّـف بسهولة للعمل على مؤول آخر بآلة أخرى ، ولهذا الهدف قمنا بإضافة مسائل بسيطة ، تجد التطبيق العملي لها على أغلب الحاسبات.

وفي النهاية ، لهؤلاء الذين يعرفون المؤول ، قمنا بإثبات الإمكانيات التي يُقدمها التأويل المشروط وإستعمال الماكرو تعليمات (MACRO INSTRUCTIONS) . ونصائح هذا الكتاب التي تدور حول البرمجة الجيِّـدة هي عبارة عن عناصر للتفكير يصبح في نهايتها البرنامج مختلفاً عن تلك المجموعة من التعليمات المبهمة كما في اللغة الثنائية . ومن الممكن إنشاء وتركيب برنامج مكتوب بلغة المؤول بشكلٍ يصبح معه واضحاً كوضوح برنامج بلغة كوبول .

لماذا جرى إختيار الحاسب IBM 370 ؟

- لأنها شاملة وعامة . وأكثر صيغ لغة المؤول العاملة عليها جرى إستعمالها وتطويرها من قبل جميع المنتجين والصانعين .

- لماضيها ومُستقبلها : إن المواصفات الخاصة بهذه اللغة والتي جهزت مع النظام IBM 360 ، قد جرت المحافظة عليها في الحاسبات IBM 370 وفي الأنظمة الجديدة من السلسلة 3000 و4000 إضافة إلى أغلب حاسبات IBM الجديدة .

القسم الأول

عموميات

1. الآلة البسيطة

هذا الفصل الأول هو مخصَّص للمبتدئين . أمّا الذي يتمتع بمفاهيم كافية تتعلَّـق بهيكل المكنة فيمكنه أن يبدأ دراسته من الفصل الثاني . إلاّ أننا نعتقد بأنه يعرض ويوضح النقاط الأساسية لعملية الفهم اللاحقة . وهو يعرِّف المصْطلحات الأساسية المتعلقة بدورة تنفيذ تعليمات الآلة .

1.1 . دراسة للآلة الحاسبة الصغيرة الجيبية

منذ النظرة الأولى ، تبدو الآلة الحاسبة الجيبية وكأنها مؤلفة من العناصر التالية :

- زر للعمل / ولوقف العمل .
- لوحة ملامس رقمية .
- شاشة للعرض .
- مجموعة من ملامس التحكم +، -، = ، . . .

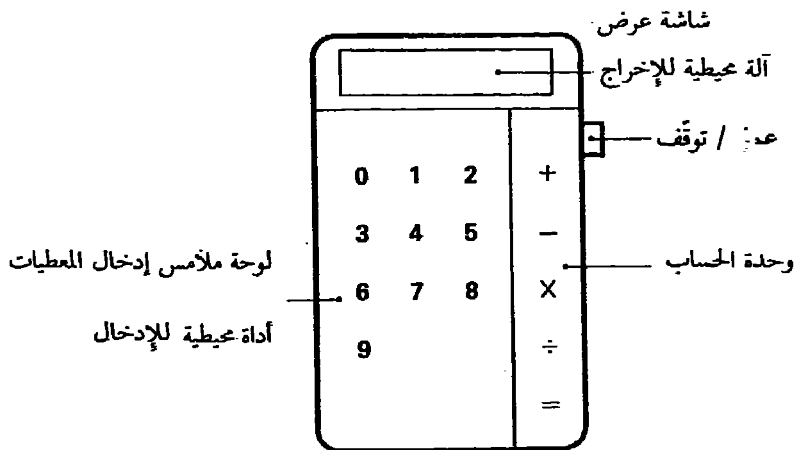
فلنقم بعملية حساب بسيطة ، القسمة مثلاً . عملية المعالجة ستجري كما يلي :

1 ـ وضع الآلة الحاسبة في العمل .
2 ـ ادخال العدد الأول (المقسوم) وعرضه .
3 ـ ضغط الزر الخاص بالقسمة.
4 ـ إدخال العدد الثاني (القاسم) وعرضه .
5 ـ الضغط على الزر = ، وعرض النتيجة .
6 ـ إيقاف عمل الآلة الحاسبة .

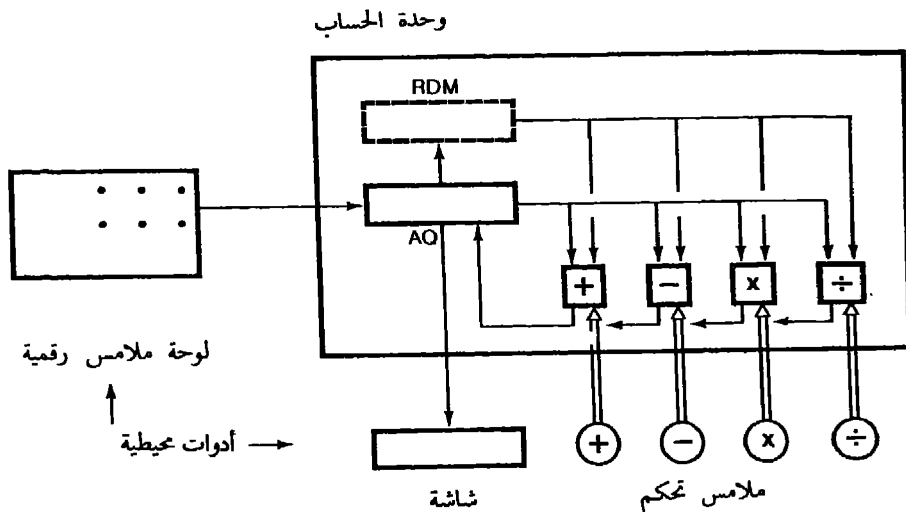
هذه السلسلة من العمليات تتطلب بعض الملاحظات :

- ترتيب العمليات هو مُحدَّد وثابت ؛ لا يمكن عكس العمليات 2 و4 .
- تتمتع مكنتنا ، إضافة إلى الدالة حساب (Compute) ، بدالّـة (مهمّـة) لإدخال المعطيات وبدالة لإخراج المعطيات (العرض على الشاشة) .
- عند إجراء العملية رقم 4 ، يختفي العدد المعروض على الشاشة ، قبل أن تتم عملية القسمة (يجب أن نعطي الصلاحية للعملية بالضغط على الململس =) ، يجب إذاً ،

وبشكل إلزامي ، أن تحتوي المكنة على ذاكرة يُخزَّن فيها العدد الأول بانتظار نهاية إدخال القاسم . فلنعرض المخطط التوضيحي[1] :



مخطط 1.1



مخطط 2.1

(1) إنّ المخطّطات المعروضة في هذا الفصل لا تدّعي تمثيل الدقّة التكنولوجية ولكنّها تعرض فقط الدّالات الأساسية المفيدة للمبرمج .

هذا المخطط يُميِّز بين نوعين من الخطوط . الخطوط البسيطة (→) والتي تُناسب خطوط إنتقال المعطيات والخطوط المزدوجة(⇐) والتي تناسب خطوط تنقل الأوامر .

تعريفات :

نسمّي وحدة حساب مجموعة دارات الجمع والطرح ، ... تُخزَّن معطيات الحساب في المناطق RDM وAQ والتي تُدعى مراصف (register) . المرصف RDM يُستخدم لتخزين العدد الأول الداخل إلى AQ للسماح بإدخال العدد الثاني .

نتيجة الحساب توضع دائماً في مرصف خاص AQ ولذلك نطلق عليه إسم مركم (Accumulator) . أمّا لوحة الملامس الرقمية وشاشة العرض فنطلق عليهما الإسم : الأدوات المحيطية للإدخال والإخراج (I/O peripherals) .

2.1 . دراسة حاسبة جيبية مع ذاكرة

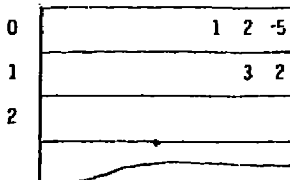
لنضف الى الحاسبة الجيبية مجموعة من خلايا الذاكرة التي سنطلق عليها الإسم : ذاكرة مركزية (Central memory) . كل خلية من الذاكرة ، وتدعى أيضاً كلمة - آلية (machine word) ، يمكنها كالمراصف أن تحتوي على مخططات أو على نتائج الحساب .
إلى كل خلية سنربط عدداً محدَّداً يُدعى عنوان الخلية ويسمح بتمييز الخلايا فيما بينها .
المؤثرات الأساسية (+ ، - ، ...) هي عبارة عن مؤثرات ثنائية (نقصد بذلك أنّها تجري بين متأثـرين (operators)) . أحد المتأثرين يكون موجوداً في المرصف AQ والآخر في المرصف RDM (مرصف معطيات الذاكرة) . كما في الحاسبات البسيطة فإن النتيجة ستكون موجودة في AQ . يصبح من الضروري أن يكون بتصرفنا :

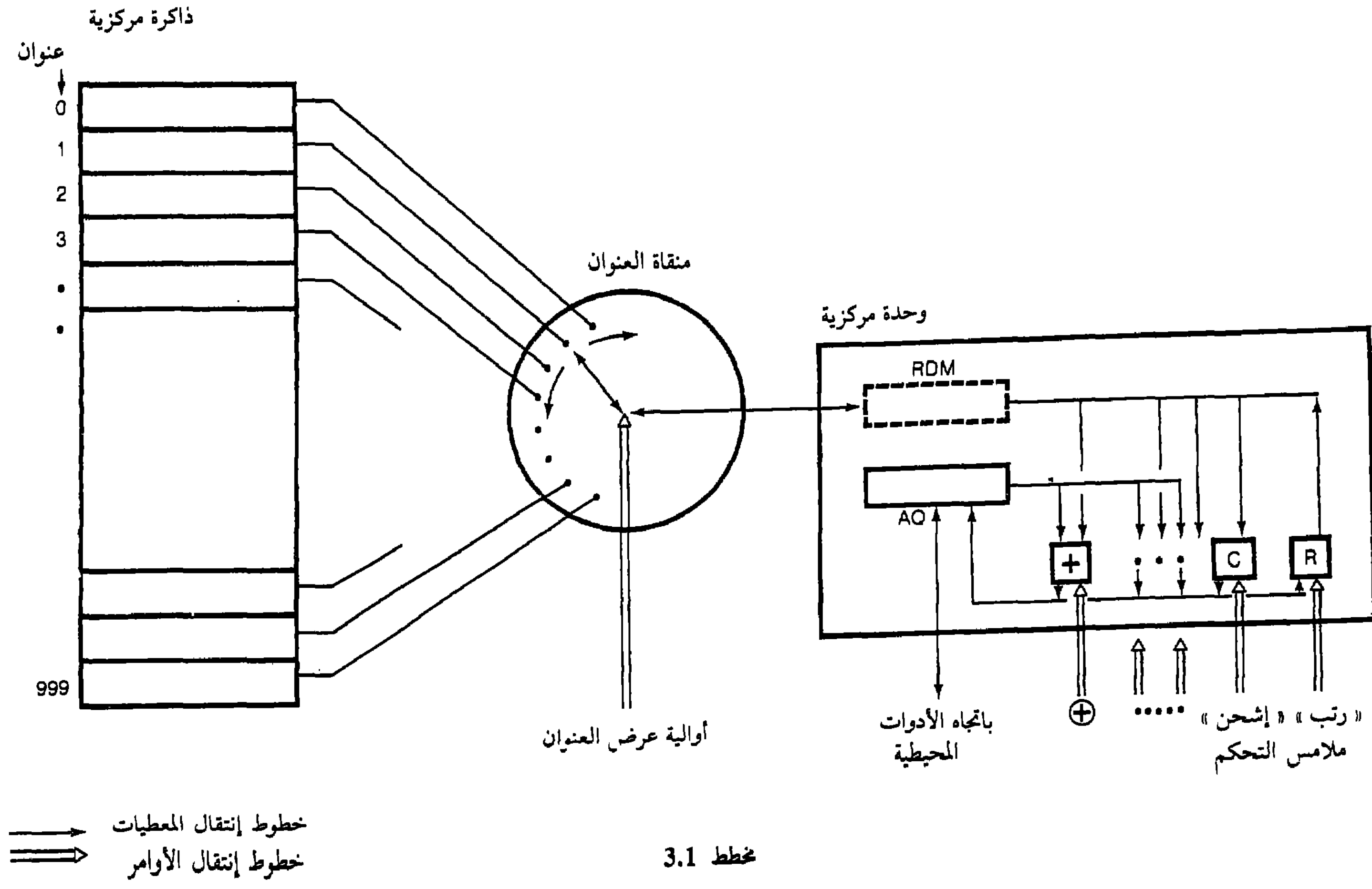
- نظام لإختيار العنوان الذي يؤمن الإتصال بين إحدى خلايا الذاكرة والمرصف RDM ؛
- دارتان إضافيتان للشحن والترتيب ، لشحن مضمون خلية من الذاكرة في المركم وترتيب مضمون المركم في عنوان معيَّـن . هكذا دارات هي موجودة على جميع الحاسبات الجيبية وتتمتع بخلية ذاكرة واحدة على الأقل . مخطط حاسبة كهذه هو ممثَّل على الشكل 3.1 .

إنّ منقاة العنوان هي هنا موضحة بواسطة ملماس دائري يؤمن الإتصال بين خلية من الذاكرة بعنوان معيَّـن ومضمون المرصف RDM . ويتعلَّق إتجاه إنتقال المعطيات بالمؤثر أو بالإشارة الحسابية المعتمدة .

مثال حول عملية حساب بسيطة .

لنفترض إن الذاكرة تحتوى على المعطيات التالية :




ذاكرة مركزية
عنوان
0
1
2
3
999
منقاة العنوان
وحدة مركزية
RDM
AQ
C
R
أوالية عرض العنوان
باتجاه الأدوات المحيطية
« رتب » « إشحن »
ملامس التحكم
خطوط إنتقال المعطيات
خطوط إنتقال الأوامر

مخطط 3.1

نرغب بجمع مضمون الخلية ذات العنوان 0 مع مضمون الخلية ذات العنوان 1 وبوضع النتيجة في العنوان 2 . فلنستعمل الترميز الكلاسيكي : (ALPHA) ، حيث ALPHA هي عبارة عن عنوان ، يشير الى مضمون الخلية ذات العنوان ALPHA . هكذا فإن (0) يعني هنا القيمة 125 . السهم سيعني إتجاه انتقال المعطيات : AQ←(0) يعني خزن مضمون الخلية ذات العنوان (0) في المركم AQ ، أي تخزين العدد 125 في AQ .

لإجراء عملية الحساب يجب :

1 ـ تركيز منقاة العنوان على 0 والضغط على الزر « إشحن » ، مما يؤدي إلى تنفيذ العملية : $(0) \rightarrow AQ$.

2 ـ تركيز منقاة العنوان على 1 والضغط على الزر + .

هذا يسمح بإجراء العملية $AQ + (1) \rightarrow AQ$. هكذا فإن هذه العملية يمكن تقسيمها إلى إثنتين .

أ) $(1) \rightarrow RDM$

ب) $AQ + RDM \rightarrow AQ$

3 ـ تركيز منقاة العنوان على 2 والضغط على الزر « خزِّن » . هذا ما يسمح بتنفيذ العملية $AQ \rightarrow (2)$.

في نهاية هذه العمليات ، ستحتوي الخلية ذات العنوان 2 على العدد 157 . والمرصف AQ يحتوي على القيمة النهائية .

ملاحظات :

جميع عمليات الحساب تتم بين المراصف AQ وRDM وليس من الذاكرة إلى الذاكرة . وهذا ما يؤدي إلى الحاجة إلى إجراء عملية شحن مسبقة للمركم .

المراصف هي إذاً عبارة عن ذاكرة مرتبطة مباشرة بدارات الحساب .

للإشارة إلى مضمون خلايا الذاكرة سنعتمد على الترميز (عنوان adresse) بشكل نستطيع معه تمييز العنوان عن مضمونه ، أي إسم «nom» الخلية وقيمتها . المراصف المذكورة لا ترد داخل أهلّة لأنه لا يوجد أي خلط ممكن بين المضمون والإسم : نعود دائماً إلى مضمون المرصف .

3.1 . من الحاسبة الصغيرة إلى الحاسب الكبير (الكومبيوتر)

إنّ كلّ معالجة تتناول معطيات وتسلسلاً دقيقاً من الأفعال ، والأوامر على الملامس + ، − ، . . . ونوع الحاسبة المعتمدة حتى الآن لا يسمح بتخزين معطيات المسألة .

الفرق الأكبر بين الحاسبة ذات الذاكرة والحاسب الكبير يكمن في كون الأخير :

- يُخزِّن ليس فقط المعطيات ولكن الأوامر المطلوب إجراؤها على المعطيات .

- يتمتع بأوالية لربط الأوامر التي ستسمح له بتنفيذ هذه الأوامر حسب الترتيب الواردة فيه . هكذا ، فذاكرة الحاسب المركزية (C.M) ستحتوي على معطيات المسألة وطريقة معالجتها للحصول على النتائج .

تعريفات :

في البداية ، سنعني كلمة أمر (Command) بالتعليمة (instruction) أو التعليمة الآلية (machine instruction) . ومجموعة التعليمات والمعطيات المرتبطة بها تؤلف البرنامج . أمّا الملامس + ، - . . . . فستختفي . ويصبح عندئذٍ من البديهي أن لا يعمل الحاسب إلا إذا كان البرنامج مسجلاً في ذاكرته المركزية .

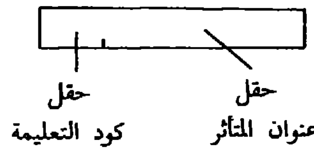
1.3.1 - هيكلية التعليمات الآلية

حسب المثل المذكور أعلاه في الفقرة 2.1 ، نستطيع أن نقول أن التعليمات الآلية هي مؤلفة من معلومتين :

1 - رقم يدل على الدارة المعتمدة من الوحدة المركزية .

2 - رقم يدل على عنوان المتأثر (Operand) .

إذا كانت التعليمة تعمل بمتأثرين (الحالة + ، - ، . . .) ، يكون المتأثر الأول مشحوناً مسبقاً في المركم (ACC) . هاتان المعلومتان ستكونان موجودتين في كلمة من الذاكرة بشكل مكوَّد رقمياً ، مثلاً حسب الطريقة التالية :



وستسمح أوالية تكويد التعليمة ، التي سنقوم بتوضيحها لاحقاً ، بكشف ومعرفة الفعل المطلوب إجراؤه على المتأثر الموجود على العنوان المذكور في التعليمة .

مثلاً :

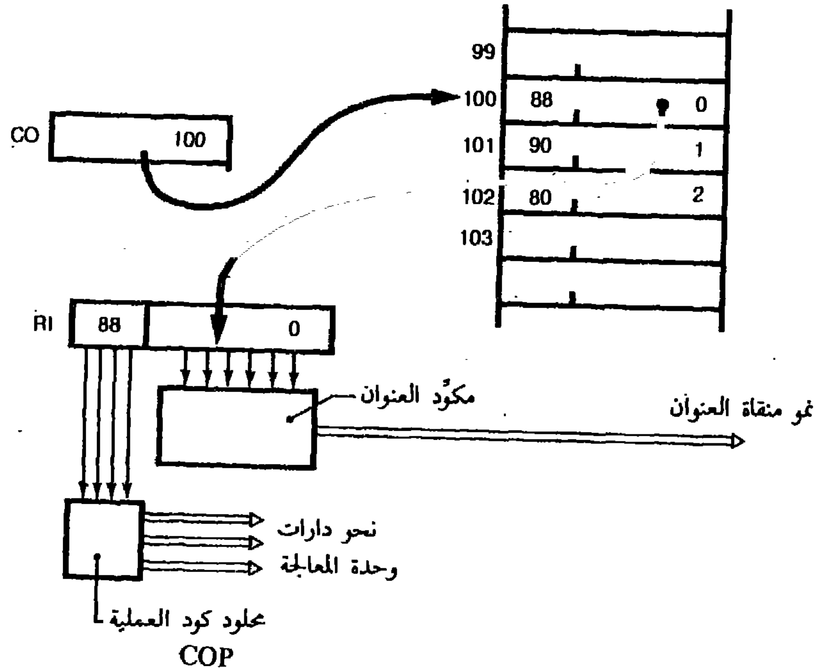
لنفترض بأن كود عملية الشحن COP هو 88 ، وإن كود الجمع هو 90 وكود الخزن هو 80 . فلنخزِّن البرنامج الذي يقوم بجمع الخليتين 0 و1 مع وضع النتيجة على العنوان 2 ، بدءاً من العنوان 100 . نحصل عندئذٍ على صورة الذاكرة التالية :

99			
100	8	8	0
101	9	0	1
102	8	0	2
103			

تنفيذ البرنامج يفترض ربطاً متتالياً للتعليمات الموجودة ، بدءاً من العنوان 100 ثمّ 101 ، . . .

2.3.1 . أوالية معرفة وربط التعليمات

تحتوي الذاكرة على نوعين من المعلومات بطبيعة دلالية مختلفة . المعطيات والتعليمات . من الضروري معاينة ومعرفة الخلية التي تحتوي على التعليمة المطلوب تنفيذها . لهذا الهدف ، هناك مرصف خاص يسمّى العدّاد الرئيسي الترتيبي (CO) أو عدّاد البرنامج program counter الذي سيحتوي في كل لحظة على العنوان التالي للتعليمة المطلوب تنفيذها . وبشكل خاص ، وفي البداية ، سيكون مشحوناً بعنوان أول تعليمة .



مخطط 4.1

منذ اللحظة التي يحتوي فيها CO على عنوان التعليمة ، فإن دورة التنفيذ تبدأ :

1 ـ إرسال التعليمة التي يشير إليها عداد البرنامج إلى مرصف التعليمة RI المرتبط بمكوِّد للعملية COP وبمنقاة العنوان .

2 ـ تكويد الغنوان الذي يقوم بتركيز منقاة العنوان ، ويحلود (يفكّ كود) COP الذي يضع الدارة المناسبة من وحدة المعالجة في حالة العمل .

3 ـ تنفيذ العملية المطلوبة بواسطة وحدة المعالجة التي ستصبح في طور العمل .

خلال المرحلة الثانية لن يكون من الضروري أن يؤشر CO على التعليمة الموجودة في طور التنفيذ ، وخلال هذه المرحلة إذاً تزداد قيمة عداد البرنامج CO واحداً (1) ليؤشر على التعليمة التالية المطلوب تنفيذها .

بعد تنفيذ التعليمة ، يعود الحاسب الى المرحلة الأولى بالقيمة الجديدة لعداد البرنامج CO وهذا يتتابع حتى نلتقي تعليمة خاصة بوقف البرنامج .

يبقى أن نشير إلى مختلف مراحل التنفيذ هي متزامنة بواسطة نبضات ساعة داخلية .

المخطط 5.1 التالي يعرض لمختلف المهام التي درسناها . وهو يشكل المخطط العملي للحاسب .

4.1 ـ خلاصة حول المكنة البسيطة

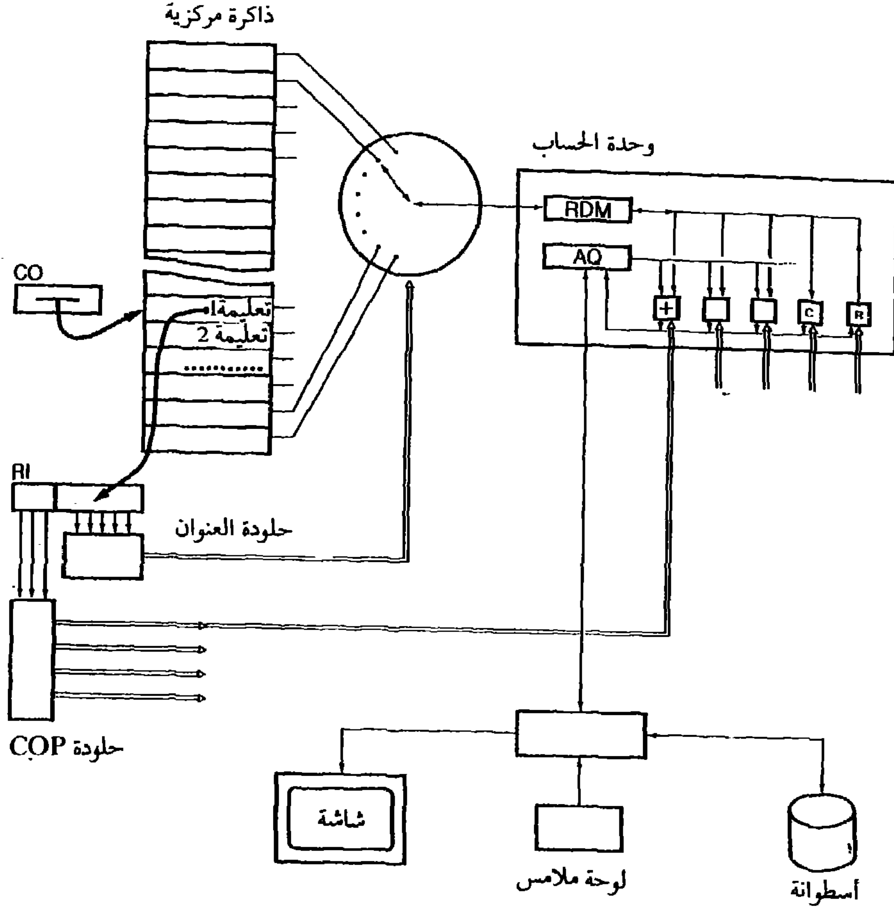
سنقوم بتوضيح الصِيغ العملية للحاسب . إنّ جميع المكنات تستعمل هذه الأواليات الأساسية ، إضافةً إلى بعض التعديلات التي سندرسها عند الحاجة . فلنحاول الآن أن نستخلص بعض الملاحظات .

ملاحظة 1

المكنة المشروحة أعلاه هي مكنة « بعنوان بسيط » ، أي أن التعليمة الآلية لا تراجع سوى عنوان واحد وإذن متأثِّر واحد علني . في هذه الحالة ، لنفترض عدداً كبيراً من المؤثرات (operators) تستعمل متأثرين والنتيجة ، ذلك يعني أن أحد المتأثّرين ثمّ النتيجة موجودان في المركم . على بعض المكنات الأخرى قد نجد تعليمات تدعى « بعنوان مزدوج » .

---

(1) عندما تكون التعليمات ذات أطوال متغيّرة (حالة الحاسبات IBM 360/370) يتقدّم العدّاد CO بمقدار طول التعليمة .



مخطط 5.1 ـ الحاسب ، المخطط العملياتي

ملاحظة 2

لا تحتوي مكنتنا سوى مركم واحد . هناك حاسبات أكثر فعالية يمكن أن تحتوي على عدد من المراصف التي تلعب دور المركم (هذه هي حالة المكنة IBM 360/370) . سيكون من الضروري أن نشير ، من داخل التعليمة ، إلى رقم المرصف الذي نعتمده كمركم .

ملاحظة 3

لنفترض ، كما في المخطط 3.1 ، أن ذاكرة المكنة تحتوي على 1000 خلية مرقمة من 0 إلى 999 .. وهذا يعني أنّ :

1 ـ عداد البرنامج يحتوي على الأقل على ثلاثة مواقع عشرية تسمح له بمراجعة جميع عناوين الذاكرة المركزية ؛

2 ـ ان حقل عنوان التعليمة ، ولنفس السبب ، يجب أن يسمح بتسجيل الأعداد من 0 إلى 999 .

ملاحظة 4

بعض التعليمات يمكن أن لا تُراجع بواسطة عنوان ما . تظهر هذه الحالة ، مثلاً ، عندما لا نستعمل سوى AQ(عكس إشارة AQ ، تصفير AQ ، الإزاحة ، . . .) . ولكن من الممكن ، عند الحاجة ، إستعمال حقل العنوان لغايات أخرى . قد يحدث ، على بعض المكنات ، أن يكون حقل العنوان مستعملاً ككود لعملية ثانوية ، مما يؤدي إلى زيادة عدد التعليمات بدون تعديل لحجم الحقل COP . أما الكود الثانوي فيُميِّز التعليمة الخاصة التي تنتمي إلى الفئة المحدَّدة بواسطة الكود الرئيسي .

ملاحظة 5

الحجم (هنا يقاس بعدد المواقع العشرية) للحقل COP يُحدِّد العدد الأقصى للدارات ـ أي للتعليمات الآلية ـ التي تراجع عنواناً وحيداً يمكن أن تحتويه وحدة الحساب .

5.1 . الحاسب ، العرض الكلاسيكي

بعد هذا المدخل ، نعود إلى عرض أكثر كلاسيكيةٍ للحاسب . لقد جرت العادة أن نُميِّز بين الأعضاء التالية :

الوحدة المركزية وتحتوي :

- الوحدة الجبرية والمنطقية (دارات عمليات ومراصف للحساب) ،

وحدة التحكم وتتألف من :

- مراصف التحكُم ،
- عداد البرنامج ،
- الساعة .

الذاكرة المركزية وتتألف من خلايا (كلمات وبايتات) معنونة ،

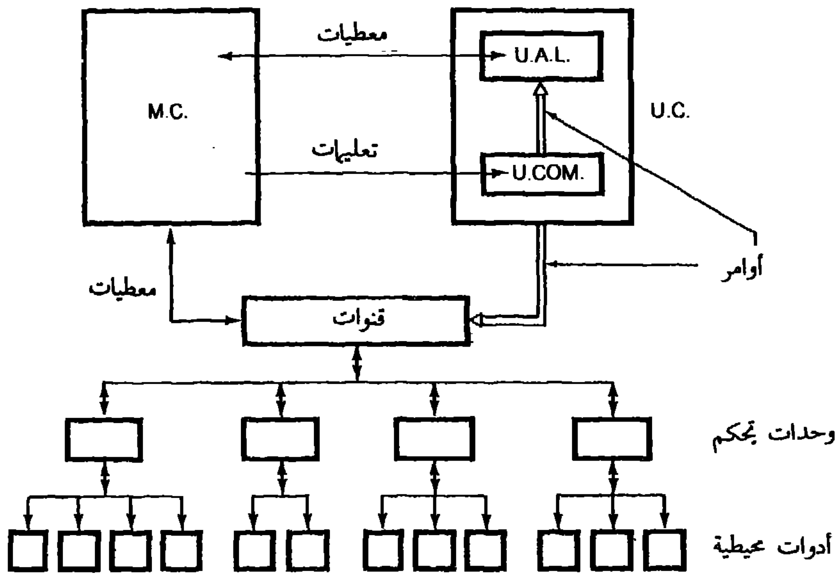
- أدوات محيطية تسمح بالإدخال والإخراج في الذاكرة المركزية للمعلومات (برامج ومعطيات) المخزَّنة على نواقل خارجية

فلنذكر البعض منها :

- قارىء البطاقات ، والمثقِّبات ، والطابعات ،
- بسَّاطة الأشرطة ، الأسطوانات والطبول المغناطيسية ،
- لوحات ملامس ، شاشات للعرض ،
- أدوات محيطية خاصة كراسم المنحنيات العاملة حسب النظام «off-line» (الاشتغال المنعزل) .

ـ القنوات أو وحدات التبادل . وهي عبارة عن الأعضاء التي ، تحت قيادة الوحدة المركزية ، تؤمن بشكل لا تزامني إنتقال المعطيات من الذاكرة المركزية إلى الأدوات المحيطية . هذه الأوالية تسمح بتحرير موارد الوحدة المركزية خلال الوقت ، نسبياً « الطويل » ، للإدخال والإخراج (I/O)[1] . التزامن بين الوحدة المركزية والقنوات (Channels) يتأمن بواسطة نظام الانقطاع الذي سنتكلم عنه لاحقاً .

ـ وحدة المراقبة والتحكم (Control unit) وهي عبارة عن أجهزة وأدوات ، متكيفة مع كل نوع من المحيطات ، وتحقّق عدداً من المهام الضرورية للإدخال والإخراج .



مخطط 6.1

(1) أعضاء الإدخال ـ الإخراج هي أجهزة الكتروميكانيكية تمثّل إذن نوعاً من القصور . إنّ قراءة بطاقة معيّنة قد تطول نحو 100 ميلّيثانية في حين أنّ وقت تنفيذ تعليمة لا يدوم أكثر من الميكروثانية μs (10^{-6} ثانية)

2 تكويد المعلومات

الإستعمال الكثير للنظام العشري جعلنا معتادين عليه ، وهذا الإعتياد جعل البعض يخشى من إستعمال نظام آخر للترقيم . ولكن تكنولوجيا الحاسبات تفرض علينا دراسة أنظمة تكويد مختلفة . يجب أن نشير إلى أن التمثيل الثنائي للمعلومات في المكنة لا يحمل أي تعديل لصيغة العمل المشروحة في الفصل الأول ، وهذا من الأسباب التي جعلتنا لا نبدأ الكتاب بهذا الفصل ، راجين أن يكون عرضنا أكثر وضوحاً .

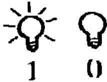
يتألف نظام التكويد من مجموعة قواعد التحويل التي تسمح بالعبور من تمثيل للمعلومات (نص فرنسي مثلاً) إلى ترميز آخر (نص بكود مورس . .) والعكس بالعكس .

الترميز الثنائي هو مفروض لأنه يسمح بتمثيل بسيط لمضمون الذاكرة والمراصف في الحاسب[1] . ويبدو أنه لترميز عدد n من حالات صمام كهربائي ، مولّع أو مطفأ ، فإن التمثيل الثنائي هو الأبسط باعتماد الاتفاق التالي :

1 ـ حالة « الضوء »

0 ـ حالة الإنطفاء

إذاً يرمز إلى الحالة بواسطة :



(1) دون الدخول في التفاصيل التكنولوجية، تمثّل المعلومات داخل الآلة بواسطة عناصر تمتلك حالتين فيزيائيتين مختلفتين .

قد نلاحظ أن مجموعة من صمامين يمكن أن تكون موجودة في عدد $4=2^2$ من الحالات المختلفة التي نرمز إليها على الشكل التالي :

0 0 حالة «0»
0 1 حالة «1»
1 0 حالة «2»
1 1 حالة «3»

ولكن بإمكاننا تكويد :
الحالة «0» : الصمّامان هما في حالة الإنطفاء
الحالة «1» : الصمام اليسار هو مطفأ ، والصمام الأيمن مولّع ، الخ

وبشكل عام ، فإن مجموعة من n من الصمّامات يمكن أن تكون موجودة في 2^nحالة مختلفة . يجب تقريب ذلك من الفعل الذي يسمح بواسطة n رقم ثنائي بأن نعدّ من 0 إلى 2^n-1 .

1.2 . أنظمة الترقيم :

لو إفترضنا أن a_i تمثّل مجموعة الرموز المستعملة لتحديد عدد بالقاعدة B ، فإن العدد الحقيقي R يُكتب على الشكل التالي :

$$\underbrace{a_n a_{n-1} \dots a_1 a_0}_{\text{القسم الصحيح}} , \underbrace{a_{-1} a_{-2} \dots}_{\text{القسم العشري}}$$

وقيمته هي :

$$R = \underbrace{a_n B^n + a_{n-1} B^{n-1} + \dots + a_0 B^0}_{\text{القسم الصحيح}} + \underbrace{a_{-1} B^{-1} + a_{-2} B^{-2} + \dots}_{\text{القسم العشري}}$$

وفي النظام العشري فإن المجموعة a_i تتألف من الرموز :

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

وفي الثنائي : 0 و1 .
وفي النظام الثماني : 0 ,1 ,2 ,3 ,4 ,5 ,6 ,7 .
وفي النظام السادس عشري (16) : 0 ,1 ,2 ,3 ,4 ,5 ,6 ,7 ,8 ,9
F, E, D, C, B, A

إنّ أسّات القاعدة B^0 ، B^1 ، B^2 ، . . . B^{-1} ، . . . تدعى أوزان الأرقام .
الجدول **1.2** يعطي قيم بعض الأوزان بالنظام العشري :

القاعدة	B^3	B^2	B^1	B^0	B^{-1}	B^{-2}
10	1000	100	10	1	0,1	0,01
2	8	4	2	1	0,5	0,25
8	512	64	8	1	0,125	0,015625
16	4096	256	16	1	0,0625	0,00390625

جدول 1.2

هكذا فالعدد 13 في القاعدة 10 يعادل $(3.10^0 + 1.10^1)$
ويُكتب على الشكل التالي : $(1.2^0 + 0.2^1 + 1.2^2 + 1.2^3)$ 1101 في النظام الثنائي .

15 في النظام الثماني : $1.8^1 + 5.8^0$
D في النظام السادس عشري : $(13.16^0$ أي $D.16^0)$
والعدد 0,75 في النظام العشري : $(5.10^{-2} + 7.10^{-1})$
يكتب : 0,11 في النظام الثنائي : $(1.2^{-2} + 1.2^{-1})$
0,6 في النظام الثماني : (6.8^{-1}) .
و 0,C في النظام السادس عشري : $C.16^{-1}$ أي 12.16^{-1} .

وفي المكنة ، تُمثّل الأعداد بشكل مكوّد ثنائياً . ويمكن أن يحتاج عدد عشري كسري إلى سلسلة طويلة ، أو لا نهائية ، من 0 و1 . وبما أن الذاكرة والمراصف لها أبعاد محدّدة عند تصميم المكنة ، لذا ، فقد يحدث تحويل عشري / ثنائي عند الحساب ، أو قد يحدث بتر لقسم من المعلومات مما يؤدي إلى فقدان الدقة في الحساب . وهذه من المشاكل التي يجب الانتباه إليها ولذا من الواجب القيام بعدد كبير من الحسابات التكرارية .

من المهم أن نلاحظ ، أنه عند إزاحة الفاصلة n موقع لجهة اليسار أو لجهة اليمين فإن هذا يؤدي إلى ضرب العدد أو قسمته على 2^n مثلاً : 13,75 يمثل بواسطة العدد 1101,11 في النظام الثنائي ، ولكن 11011,1 يعادل 27,5 و110,111 يعادل 6,875 .

عشري	ثنائي	سادس عشري	ثماني
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	8	10
9	1001	9	11
10	1010	A	12
11	1011	B	13
12	1100	C	14
13	1101	D	15
14	1110	E	16
15	1111	F	17

جدول 2.2

2.2 . تغيير القاعدة

سنترك للقارىء أن يعود للمراجع إذا رغب بذلك . وسنذكر ، بواسطة بعض الأمثلة ، إن التحويلات الثنائية / الثمانية والثنائية / السادس عشرية هي متزامنة لأن القواعد 8 و16 هي عبارة عن أسّات صحيحة للقاعدة 2 .

ينقلب العدد الثنائي إلى سادس عشري بدءاً من كل جهة من موقع الفاصلة وبتقطيع العدد إلى أقسام مؤلفة من أربعة أرقام ثنائية أو بتات[1] وبتأويل كل قسم :

ثنائي 1001101011,11001
سادس عشري 2 6 B , C 8

الرقم الأخير «8» نحصل عليه بتوسيع الرقم 1 بوضع أصفار لجهة اليمين .

التحويل الثنائي / الثماني يتم بتقطيع العدد الثنائي إلى أقسام مؤلفة من ثلاثة أرقام . نحصل عندها على 1153, 62 في النظام الثماني .

التحويل المعاكس هو بديهي .

3.2 . الفائدة من النظامين السادس عشري والثماني

سنرى أن كل كلمة آلية هي مكوّنة من عدد متحول ، يتعلّق بالحاسب ، من العناصر التي تدعى بتات[1] (bit) . كل عنصر يمكن أن يكون موجوداً ، كما هي الحالة

(1) من BIT وهو اختصار للمصطلح الأميركي BInary digiT ، أي رقم ثنائي .

بالنسبة للصمّام ، في واحدة من حالتين فيزيائيتين ، لذا يصبح من الطبيعي ترميز حالة البتة بواسطة 0 أو 1 ومضمون الكلمة ـ الآلية ؛ ليس كما في الفصل الأول بواسطة رقم عشري ،.بل بواسطة سلسلة من الأرقام 0 أو 1 ، ويمكن تفسير مجموعة البتات كعدد ممثل في النظام الثنائي .

الأحجام ، المحدَّدة بعدد البتات ، للكلمات ـ الآلية التي نلتقيها عادة في الحاسبات هي بطول 8 (الميكروبروسسور) ، 16 ، 24 ، 32 (IBM 360/370) ، 36 ، 48 و 60 بتة . عند تمثيل مضمون كلمة ـ ذاكرة على ورقة فهذا يتطلب من 16 إلى 60 رمزاً . التمثيل السادس عشري والثماني يظهران إذن مفيدين مهمّين كثيراً لأنهما يُقسّمان على 4 أو على 3 عدد الرموز المطلوب كتابتها وذلك مع المحافظة على إمكانية تحويلها فوراً إلى النظام الثنائي . ولكن النسخ اليدوي لعدد محدَّد بالنظام السادس عشري هو منبع لعدد أقل من الأخطاء منه في حال كتابته في النظام الثنائي . لذلك فللقارىء فائدة من الإعتياد على هذا النوع من التمثيل المعتمد لتمثيل المعلومات في الذاكرة .

4.2 . الحساب في النظامين الثنائي والسادس عشري

لن نقوم سوى بإعطاء بعض الأمثلة التي يجب أن تسمح للقارىء بإجراء بعض العمليات البسيطة بالجمع والطرح .

في النظام الثنائي :

0	0	1	1
+0	+1	+0	+1
0	1	1	10

مثلاً :

```
 1101
 +111
10100
```

في النظام السادس عشري :

```
 D
+7
14
```

في النظام السادس عشري من العملي تحويل كل رقم الى النظام العشري ، وإجراء العملية في هذا النظام ومن ثم تحويل النتيجة . مثلاً :

$D_{16}=13_{10}$, $7_{16}=7_{10}$, $13+7=20_{10}=16+4$: نضع 4 ونحفظ باليد 1 ، إلخ :

```
  3F2     3F2
 +1A4    -1A4
  596     24E
```

بنفس الطريقة نقوم بإجراء الطرح 4 − 2_{16} تصبح 4−16+2 أي E وباليد 1 . . .

حسب نفس الصيغة سنستطيع إجراء الحساب في النظام الثماني . وباستطاعة القارىء أن يتمرّن بوجود الأمثلة المعطاة في نهاية الفصل .

5.2 . التمثيل الداخلي للمعطيات

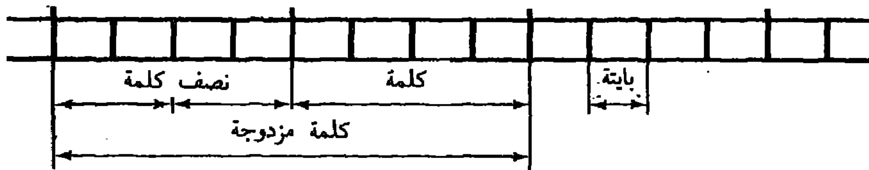
1.5.2 . الذاكرة

حتى هذا الوقت إعتبرنا إن الذاكرة هي مؤلفة من خلايا مرقّمة بدءاً من 0 ، الخلية هي الكلمة ـ الآلية والعناوين هي عناوين الكلمات .

سنقوم بتحديد الأشياء . المكنات IBM 360/370 تتمتع بكلمة ـ آلية من 32 بتة مرقّمة من اليسار إلى اليمين من 0 إلى 31 . تُقسَّم الكلمة إلى أربع بايتات (تشكيلة من 8 بتات) . والبايتة هي قابلة للعنونة . سنتكلّم عن الذاكرة المعنونة بالسمات (وسنرى إن السمة قابلة للتمثيل بواسطة 8 بتات) مقابلة مع بعض المكنات حيث الذاكرة معنونة بالكلمات . عنوان الكلمة هو إذاً عنوان البايتة الأولى من الكلمة . في النهاية نوجز ما يلي :

- جبهات النصف كلمات هي بعناوين مزدوجة ؛
- جبهات الكلمات هي بعناوين قابلة للقسمة على أربعة ؛
- جبهات الكلمات المزدوجة تتمتع بعناوين قابلة للقسمة على 8 ؛

ومع إن الذاكرة هي قابلة للعنونة في مستوى البايتة ، يجب السهر على المحافظة على هذا التقسيم للمعطيات الممثلة بواسطة نصف كلمة ، كلمة ، أو كلمة مزدوجة .



شكل 3.2

2.5.2 . تمثيل المعطيات اللارقمية

بإمكاننا تكويد نوعين من المعلومات في الذاكرة : المعطيات الرقمية والتي هي عبارة عن تشكيلات ثنائية مرتبطة بمعنى رقمي ، والمعطيات من نوع سمات والمعالجة كوحدات غير رقمية .

لقد كان من الملائم عند تصوُّر مكنات IBM 360/370 ، تكويد السمات بواسطة 8 بتات . هذا النظام يسمح بتكويد 2^8 ، أي ما مجموعه 256 كوداً مختلفاً . هذا التصوُّر هو واسع الإنتشار ، ولكن هناك مكنات أخرى تستعمل تكويد السمات بواسطة 6 بتات -تُحدِّد مجموعة السمات المتوفِّرة بالعدد 64 سمة .

قد يبدو لنا مفاجئاً إعتماد كود لتمثيل السمات بواسطة 8 بتات . فلنلاحظ ببساطة إن هذا النظام يسمح لنا بالحصول على ألفباء واسعة تحتوي على السمات الكبيرة ، والصغيرة ، والسمات العشر العشرية وبعض السمات الخاصة ، كإشارات العمليات ، وعلامات الوقف ، والفسحة ، الخ .

الكود الداخلي لتمثيل السمات ، والمستعمل على المكنات IBM 360/370 هو EBCDIC (Extended Binary Coded Decimal Interchange Code) . يُرمز إلى الحرف A بواسطة الكود 11000001 ، أي C1 بالترميز السادس عشري . ويكوِّد الحرف «B» بواسطة C2 وهكذا دواليك . لائحة الأكواد موجودة في الملحق .

مثلاً : لنفترض إن مضمون حيِّز الذاكرة هو التالي :

0 0	C 1	E 2	E 2	C 5	D 4	C 2	D 3	C 5	E 4	D 9	4 0	F 3	F 7	F 0	0 0
	100				104				108				112		

تأويل هذه السلسلة من 14 بايتة ، والتي تبدأ بالعنوان 100 ، هو حسب الكود «ASSEMBLER 370» .

نشير إلى وجود علاقة تراتبية بين القيم الثنائية المستعملة للتكويد :

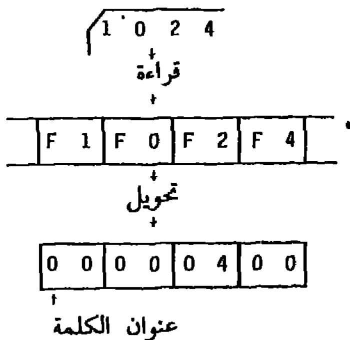
$$40 < C1 < C2 < \ldots < F0 < F1 < \ldots < F9$$

وهذا ممكن أن يترجم بواسطة :

كود الأرقام < . . . < كود B < كود A < كود القسمة .

هذه الخصوصية هي مستعملة للترتيب الأبجدي .

يجب أن نُميِّز بين التمثيل الأبجعددي والتمثيل الرقمي . المثل التالي يُوضح لنا التحويل المعتمد لمعطى مقروء من البطاقة ومحوَّل إلى ثنائي .



ناقل خارجي

ذاكرة (حيِّز إدخال)
تمثيل بطريقة السمات

ذاكرة (متحولة مؤشرة في FORTRAN في لائحة الأمر READ) .
تمثيل بطريقة الفاصلة الثابتة

التمثيل السياقي يُقال عنه أيضاً « القابل للتنقيح » لأنه ضمن هذا الشكل يجب أن تكون المعلومات موجودة قبل أن تستلمها الطابعة لطبعها .

3.5.2 . تمثيل المعطيات الرقمية

المعتادون على لغة فورتران يعلمون أن المتحولة أو الثابتة يجب أن تُمثَّل دائماً في المكنة بواسطة كلمة (أو كلمة مزدوجة عندما يكون الحيِّز مصرَّحاً عنه بدقة مزدوجة) . ويعلمون أيضاً إن هذه اللغة تستعمل نوعين أساسيين من التمثيل الداخلي للمعطيات الرقمية : النوع الصحيح (integer) والنوع العائم (real) .

أما المعتادون على لغة كوبول فلا يجهلون ان الحسابات الجارية بهذه اللغة تتم بواسطة تمثيل مجهول من لغة فورتران : التمثيل العشري المتراص . سنجد هذه الطرق الأربع في تكويد الأعداد في مستوى المكنة : الطريقة «الفاصلة الثابتة (fixed point) » (صحيح بلغة فورتران) ، والعائم البسيط والعائم الموسّع والصيغة العشرية المتراصّة . نشير إلى أنّ مع كلّ نوع من هذه التمثيلات تتلاءم مجموعة من المؤشّرات (دارات الكترونية ، + ، − ، . . .) ، صالحة للعمل بهذه التشكيلات الثنائية . وفي النتيجة فإن المكنات تحتوي على أربع مجموعات من التعليمات الجبرية .

أ ـ التمثيل بفاصلة ثابتة

بهذه التسمية يجب أن نفهم « فاصلة ثابتة إتفاقياً » . هكذا ، فالفاصلة ، عنصر أساسي من قيمة العدد ، لا تظهر أبداً في التمثيل الداخلي للعدد في الذاكرة . ولقد لاحظنا (في الفقرة 1.2) إن التشكيلات الثنائية المعتمدة لـ n ، 2n وn/2 لا تختلف إلا بواسطة موقع الفاصلة ، لذا ، فإن 1001 يمكن أن تُمثِّل القيمة 9 إذا إعتبرنا إن الفاصلة موجودة لجهة اليمين ، أو 0,5625 إذا إعتبرنا إن الفاصلة موجودة في أقصى اليسار النظام IBM 360/370 يفترض الفاصلة موضوعة لجهة اليمين . وللتأكد من ذلك يكفي ملاحظة التعليمات التي تسمح بجمع المعطيات بطول مختلف (كلمة أو نصف كلمة) . إن عملية التسطير للمعلومات تتم لجهة اليمين . هذا التمثيل هو إذاً التمثيل الصحيح . وهناك بعض المصممين الآخرين الذين إعتمدوا الإتفاق المعاكس ، أي الفاصلة لجهة اليسار .

تُكوَّد الأعداد حسب النظام الثنائي في كلمة ـ آلية . البتة ذات الوزن الأكبر (البتة الموجودة لجهة اليسار) ترمز إلى الإشارة الحسابية . إذا كانت تساوي 0 ، يكون العدد إيجابياً ، أما إذا كانت تعادل 1 فمعنى ذلك أن العدد هو سلبي .

بواسطة n بتة باستطاعتنا تعداد من 0 حتى 2^{n-1} . وإذا حجزنا بتة للإشارة فسيكون بإمكاننا تمثيل الأعداد الصحيحة I بحيث إنّ :

$$-2^{n-1} \leq I \leq 2^{n-1}-1$$

إذا كانت $n = 16$: $-32768 \leqslant I \leqslant +32767$

تمثيل الأعداد الإيجابية لا يفترض أية مشكلة ، والتأويل العشري نحصل عليه بضرب كل بتة بالوزن المعتمد للموقع . وفي المقابل يجب أن نعتمد إتفاقاً جديداً للأعداد السلبية .

تمثيل الإشارة والقيمة المطلقة

الفكرة التي تخطر لنا تقوم على إعتبار البتة ذات الوزن الأقوى ترمز إلى الإشارة والباقي يرمز إلى القيمة المطلقة للعدد . حسب هذا الإتفاق ، الممثل بأربع بتات :

5 + يكتب :	0101	
5 − يُكتب :	1101	
نتيجة الجمع :	10010	هذه النتيجة هي ليست حقيقية .

هذا التمثيل يُحتم علينا إذاً ، للحصول على النتيجة الصحيحة ، أن نفحص الإشارات المرتبطة بالمتأثرات قبل إجراء العمليات . لا يجب معالجة الأعداد السلبية والإيجابية بنفس الطريقة . يمكن للقارىء أن يقتنع بأن إعتماد هذه الصيغة يحتم علينا إعتماد منطق ألكتروني أكثر تعقيداً . وقد جرى التخلي عنه اليوم .

التمثيل المدعو مُكمِّـل 1 (1 Complement)

عكس العدد (أو ضدّه) . نحصل عليه بأخذ عكس كل بتة . بما فيها بتة الإشارة . هكذا :

5 + تكتب :	0101
5 − تُكتب :	1010
	1111

وبنتيجة الجمع نحصل على

أي ، أن مُكمِّـل 1 هو 0000

من الممكن إعتبار إن هذا النوع من التمثيل يؤدي إلى إدخال 0 إيجابي .وصفر سلبي . المسائل المطروحة في نهاية الفصل تشرح سيئات هذه الاتفاقات وفوائد الاتفاقات اللاحقة

التمثيل المدعو « مُكمِّـل إلى 2 » (Two complement)

هو التمثيل المعتمد على المكنات IBM 360/370 . يُمثّـل كل عدد سلبي بواسطة المكمِّـل إلى 2^n لعكس العدد . ولو إفترضنا إن X هو العدد ، وأن $\bar{X}$ هو مُكمِّـل العدد X إلى 2^n ، نحصل إذاً على العلاقة التالية $X + \bar{X} = 2^n$. الإتفاق حول الإشارة هو كالسابق . ونشير إلى أن المعطيات الرقمية هي مكوّدة بأطوال ثابتة ، هي الكلمات ـ

الآلية . وللمكنات IBM 360/370 ، n تعادل 32 . ولتسهيل العمل ، فإننا سنعالج مسائل تعمل بأربع أو ثمان بتات .

وبالتكويد بواسطة أربع بتات ، حيث البتة اليسرى هي بتة الإشارة ، فإنّ كود العدد 5- هو المُعادل الثنائي لـ $2^4-5=11$ إذن :

+5	0101
−5	1011
0	10000

وبإهمال الحاصل بعد موقع الإشارة نحصل على صفر .

الطريقة للحصول على المُكمِّل إلى 2 لعدد ما تكمن بتكملة العدد إلى 1 وبعد ذلك إضافة 1 إليه . تتم العمليات على جميع البتات بما فيها بتة الإشارة .

مثلاً :

	+5	0101
مكمِّل إلى	1	1010
	+1	+1
		1011

فلنلاحظ إنه إذا كنا نعمل على عدد ثنائي مُمثَّل بالترقيم السادس عشري ، فإن المُكمِّل إلى 2^n يصبح مُكمِّلاً إلى 16^p سنحصل على التمثيل السادس عشري للعدد المعكوس بتكملة كل رقم إلى F وإضافة 1 .

مثلاً : على ثمان بتات :

0100 1101	→	4D
		↓
		B2
		+1
1011 0011	←	B3

إنتقال العدد ، المُمثَّل بواسطة 16 بتة ، في مرصف بطول 32 بتة سيتم بواسطة إنتقال بسيط إلى اليسار للبتة ذات الوزن الأكبر :

0A1C	→	0000 0A1C
B0D3	→	FFFF B0D3

حالة الفيض عن السعة (Over flow) ، يمكن أن تحدث عند إجراء عملية معينة وذلك عندما يكون كلا المتأثرين بنفس الإشارة والنتيجة تصبح بإشارة معاكسة .

لنعطِ بعض الأمثلة على معطيات ممثلة بواسطة أربع بتات . مجموعة الأعداد القابلة للتمثيل هي :

		0000	0
1111	-1	0001	+1
1110	-2	0010	+2
1101	-3	0011	+3
1100	-4	0100	+4
1011	-5	0101	+5
1010	-6	0110	+6
1001	-7	0111	+7
1000	-8		

بالإمكان إهمال المرحِّل لليسار بدءاً من موقع الإشارة ، إذا كان كلا المتأثرين بنفس الإشارة ، والنتيجة بنفس الإشارة .

وجود المُرحِّل ، ويُدعى (Carry) في المصطلحات الأنكلوسكسونية ، ليس هو إشارة خطأ في الحساب .

سنلاحظ في النهاية إن العدد الأصغر القابل للتمثيل هو -2^{n-1} والأكبر هو $2^{n-1}-1$ وإن الطرح يمكن أن يتم بواسطة الجمع إلى مُكمِّل 2 .

عشري	مُرحِّل مفقود		نتيجة
+7 +7 = 14		0 111 + 0 111 = 1 110	خطأ DDC (1)
+4 +5 = 9		0 100 + 0 101 = 1 001	خطأ DDC (1)
+4 -5 = -1		0 100 + 1 011 = 1 111	صحيح
-4 -5 = -9	1	1 100 + 1 011 = 0 111	خطأ DDC (1)
-3 +3 = 0	1	1 101 + 0 011 = 0 000	صحيح

(1) DDC : الفيض عن السعة (over flow

ب ـ التمثيل بفاصلة متحركة

الحساب العلمي يستعمل عادة أعداداً بأحجام كبيرة جداً أو صغيرة جداً ولكن ممثلة بواسطة عدد محدّد من الأرقام . النوع فاصلة ثابتة لا يسمح بالتمثيل البسيط لهذه الأعداد ، ولذلك إعتمدنا طريقة أخرى في التكويد المركّب من قسمين :

- المميّزة (الأسّ المعيّن) التي تعطي الحجم .
- القسم العشري (mantisse) الذي يحدد الأرقام ذات الأوزان الكبرى .

هكذا فبإمكاننا تحديد العدد على الشكل التالي :

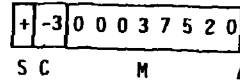
$$S.M.B^{C}$$

حيث S هي الاشارة ، M القسم العشري (mantisse) ، وB عدد ثابت (2 ، 10 ، أو 16 حسب المكنة) ، C هي الأس المعيّن .

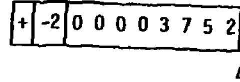
كما في الفاصلة الثابتة ، فإن الفاصلة لا تظهر في التكويد الداخلي ولكنها توضع عادة إلى يمين أو إلى يسار القسم العشري M . هكذا ، فلنظام بقاعدة B=10 ، يُكتب العدد 37,52 على الشكل التالي[1] :

1 ـ الفاصلة لجهة يمين القسم العشري .

(1) 37520.10^{-3}

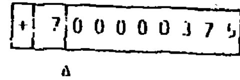


(2) 3752.10^{-2}

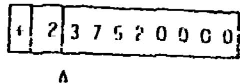


2 ـ الفاصلة إلى يسار القسم العشري ،

(3) $0,000003752.10^{7}$



(4) $0,3752.10^{2}$



نلاحظ ، في الحالة التي تكون فيها الفاصلة موجودة إلى يسار القسم العشري ، إن التمثيل (4) يعطي عدداً أكبر من الأرقام ذات المعنى (Significants digits) من التمثيل

---

(1) Δ : رمز يشير إلى موقع الفاصلة .

(3) ، في الحالة التي يكون فيها عدد الأرقام المحجوزة للقسم العشري ثابتاً . التمثيل (4) يُدعى موحد التنظيم المعاير (normalized) . وهو يتناسب مع حصر الأرقام ذات المعنى من القسم العشري لجهة اليسار . هذا التمثيل يسمح بأكثر دقة ممكنة .

من الممكن أن نعبر من تمثيل معيَّـن إلى تمثيل معاير آخر بواسطة إزاحة الأرقام وتعديل الأس .

إذا كانت B = 10 ، فإن الإزاحة الى اليسار لموقع رقم يؤدي إلى تنقيص الأس المعيَّـن 1 .

أما إذا كانت B = 16 ، فإن الإزاحة إلى اليسار لرقم سادس عشري من القسم العشري سيؤدي إلى تنقيص 1 من الأس المعيَّـن . بهكذا سيكون العدد ممثلاً بشكل معاير عندما لا يكون الرقم السادس عشري ذو الوزن الأكبر من القسم العشري صفراً . سنشير إلى أن الإزاحة لموقع سادس عشري يُترجم إزاحة أربعة مواقع ثنائية . على الحاسبات IBM 360/370 :

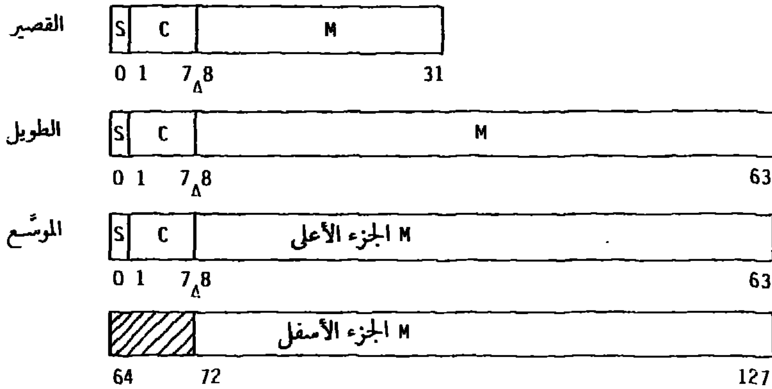
- الإشارة S من العدد هي مكوَّدة على بتة واحدة (+ = 0 ، − = 1) ؛
- القاعدة B هي 16 ؛
- يفترض أن تكون الفاصلة إلى يسار القسم العشري الذي يُمثِّـل عدداً أصغر من 1 .
- العدد الثنائي المكوَّد في الحيِّـز C بطول 7 بتات والمحفوظ للأس المعيَّـن ، لا يُمثِّـل أبداً قيمة الأس المعيَّـن E لـ 16 ولكن :

$$C = 64_{10} + E$$

لذا فهناك مشكلة في تكويد إشارة الأس كي نحصل على قوى سلبية وإيجابية للقاعدة B بدلاً من اعتماد ترميز شبيه بالمُكمِّـل إلى 2 ، لقد جرى إختيار اعتماد القوة صفر في التكويد المناسب للقيمة الوسطية للأعداد القصوى 0 و1− 2^7 أي 64_{10} أو 40_{16} أو 1000000_2 . هكذا ، عندما تكون $C=64_{10}$ فإن قيمة العدد هي $S.16^0.M$ ، وعندما تكون $C>64$ $(C=64+E)$ فإن قيمة العدد هي $C.S.16^E.M.$ متغيِّـرة من 0 إلى 127 وبالتالي $-64 \leq E \leq +63$.

للحصول على E يكفي ، في النظام السادس عشري ، أن نطرح 40_{16} : 46_{16} تناسب E=6 و$3F_{16}$ تناسب E = − 1 .

يوجد على الحاسبات IBM 370 ثلاثة أشكال بفاصلة متحركة . الأعداد بالفاصلة المتحركة الصغيرة تحتل كلمة ـ آلية ، والأعداد الطويلة تحتل كلمتين ـ آليتين ، والأعداد الموسَّـعة تشغل أربع كلمات . الشكل الأخير هو غير موجود على المكنات 360 .



مخطط 4.2

الأشكال الثلاثة تسمح بتكويد أعداد بنفس الحجم . وتختلف بواسطة عدد الأرقام ذات المعنى التي تقدّمها . العدد P هو :

بالشكل القصير :

$$16^{-65} \leqslant P \leqslant (1-16^{-6}).16^{63}$$

7 أرقام عشرية ذات معنى

- بالشكل الطويل

$$16^{-65} \leqslant P \leqslant (1-16^{-14}).16^{63}$$

16 رقماً عشرياً ذا معنى .

- بالشكل الموسع

$$16^{-65} \leqslant P \leqslant (1-16^{-28}).16^{63}$$

33 رقماً عشرياً ذا معنى .

وفي الحالات الثلاث يكون معنا تقريباً : $5,4.10^{-79} \leqslant P \leqslant 7,2.10^{75}$

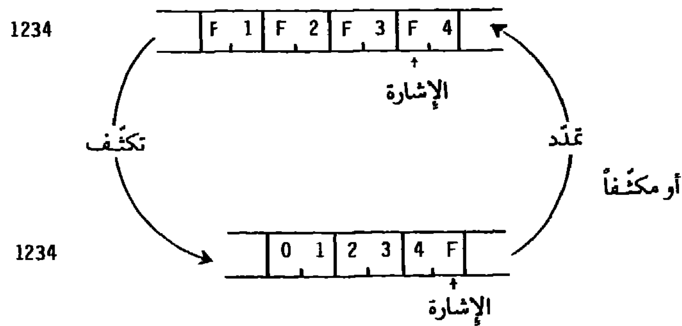
أما الحسابات بواسطة هذه الطرق في التمثيل فقد تؤدي إلى فيض عن السعة (Overflow) عندما نحصل على قِيم كبيرة جداً أو صغيرة وتدعى Overflow أو Underflow للأعداد بالفاصلة المتحركة .

مثلاً : التمثيل بفاصلة متحركة

C 2 1 9 0 0 0 0	-25	$-(1.16^{-1}+9.16^{-2})\ 16^{2}$
C 1 1 0 0 0 0 0	-1	$-(1.16^{-1})\ 16^{1}$
0 0 0 0 0 0 0 0	0	$0\ .\ 16^{-64}$

ج - التمثيل العشري

يمكن أن يتم تمثيل العدد بواسطة النظام العشري المكوَّد ثنائياً (DCB) موسّعـاً ، أي على شكل سمات .



هذا التمثيل ، الواسع الإنتشار في الإدارة ، هو أقل « تراصاً » من سوابقه . لا يوجد أي طول ضمني لها : توضع المعطيات بداخل بايتات . سنرى ان التعليمات PACK وUNPK تسمح بالعبور من شكل إلى آخر .

الفاصلة ، كما رأينا ، ليست مُمثَّلة . وإن تنظيم موقعها والاصطفاف المحتمل المناسب يقع على عاتق المبرمج . ونشير إلى المواقع المختلفة للإشارة . القِيم السادس عشرية A, C, F, E يجري تأويلها وكأنها « + » . أمّا B وD فيؤوّلان وكأنهما « - » .

تمارين

تمرين 1.2 ـ غيّـر إلى النظام الثنائي والسادس عشري ، الأعداد العشرية التالية :
15 35 256 1024 348,5 .

تمرين 2.2 ـ غيّـر إلى النظام العشري والثنائي الأعداد السادس عشرية التالية :
ABC 1A3B FFF 3A

تمرين 3.2 ـ إحسب المكمِّـل إلى 2 للعدد 1A3B . أطرح 1A3B من العدد 2ABC .
أعطِ التمثيل الموسّـع إلى 32 بتة للعدد 1A3B وكذلك لمكمّـله إلى 2 .

تمرين 4.2 ـ أعطِ القيم الرقمية العشرية التي نقوم بتأويلها :
C1F00000

ـ كمعطى ممثّـل بفاصلة ثابتة حسب تكويد الاشارة والقيمة المطلقة .
ـ كعدد ممثّـل بالمكمِّـل إلى 2 .
ـ كعدد بفاصلة متحركة بطول قصير (هل هو معايَر في هذه الحالة؟) .
هل بالإمكان اعتبار هذا التشكيل الثنائي كمعطى مكوّد بالشكل العشريّ ؟

ما هو نقيض (أو ضدّ) هذا العدد في كلّ من التمثيلات المذكورة ؟

تمرين 5.2 ـ عاير العدد بفاصلة متحركة C5032000 .

3 . العنونة المطلقة ، العنونة النسبية

1.3 . عموميات

في الفصل الأول عرضنا التعليمات ـ الآلية وكأنها مشكّلة من حقلين : الحقل كود العملية (operation code) وحقل العنوان . تحتوي التعليمة إذاً على العنوان المطلق للمتأثر ، أي عنوانه الفعلي أو الحقيقي بالنسبة للعنوان 0 من الذاكرة . هكذا فبرنامج جمع مضمون الخلايا 0 و1 وخزن النتيجة في العنوان 2 كان قد كُتب على الشكل التالي:-

0			المتأثر الأول
1			المتأثر الثاني
2			النتيجة
3	8 8	0	(0) → AQ
4	9 0	1	AQ+1 → AQ
5	8 0	2	AQ → (2)

فلنفترض بأننا زرعنا هذا البرنامج (مجموعة مناطق العمل والتعليمات) ليس على العنوان 0 ولكن على العنوان 100 . سنكتب عند ذلك :

100			المتأثر الأول
101			المتأثر الثاني
102			النتيجة
103	8 8	1 0 0	(100) → AQ
104	9 0	1 0 1	AQ+(101) → AQ
105	8 0	1 0 2	AQ → (102)

نلاحظ أن كود العمليات لا يتغيّر ولكن العناوين قد جرى نقلها 100 موقع لأن التعليمات تعود إلى العناوين المطلقة . أو بشكل آخر ، فإن كتابة البرنامج تتعلّق بالعنوان الفعلي لمكان البرنامج . هذا الإلزام ، الذي سنعرض سيئاته ، قد أجبر مُصمّمي المكنات على تعريف أوالية العنونة النسبية : حقل العنوان من التعليمة لا يعود إلى العنوان المطلق للمتأثر ولكن إلى عنوان نسبي حسب عنوان أساسي (مطلق) . وبالإجمال فإن حقل العنوان يعطي « المسافة » إلى موقع المتأثر بالنسبة إلى عنوان يُعتبر وكأنه أساس أو قاعدة (Base adresse) ويعرَّف في لحظة زرع البرنامج في الذاكرة . العنوان الفعلي (المطلق) للمتأثر سيحسب ، في لحظة تنفيذ التعليمة ، بواسطة جمع العنوان المرجعي (الأساسي) إلى قيمة الإزاحة المحدَّدة في التعليمة .

سنعمد في ما يلي إلى شرح أواليات عدّة للعنونة تتواجد في نفس الوقت على الآلات الحالية .

2.3 . العنونة القاعدية

هي عنونة نسبية حيث المبدأ هو كما ورد أعلاه . يحتوي الحاسب على عدد من المراصف التي يمكن أن تستعمل كمراصف أساسية (قاعدية) ، ويجب على المبرمج :

- أن يختار المرصف الأساسي بواسطة أمر خاص .
- أن يُخزِّن قيمة معينة في هذا المرصف ، قيمة ستكون عبارة عن العنوان الأساسي .
- كتابة البرنامج (معطيات وتعليمات) نسبة إلى عنوان معين يعادل عادة الصفر.

وفي لحظة التنفيذ يُشحن البرنامج في الذاكرة ، وتُخزَّن قيمة العنوان القاعدي في المرصف القاعدي . عند تنفيذ كل تعليمة فإن العنوان الموجود في التعليمة (الإزاحة déplacement) يُضاف أوتوماتيكياً إلى مضمون المرصف القاعدي للحصول على العنوان الفعلي للمتأثر .

ذاكرة

المتأثر الأول		150
المتأثر الثاني		151
النتيجة		152
	8 8 0	153
	9 0 1	154
	8 0 2	155
		156

المرصف القاعدي: 1 5 0

يُكتب البرنامج دون الإهتمام بالعنوان الفعلي لمكان خزن البرنامج . وتُحسب جميع العناوين نسبةً إلى العنوان صفر (بداية البرنامج) .

ولنفترض إن بداية البرنامج (العنوان النسبي صفر) موجودة على العنوان الفعلي 150 ، وهي قيمة سيتم خزنها في مرصف القاعدة[1] . إذاً فالعنوان النسبي n للبرنامج يناسب العنوان الفعلي n + 150 . . . والبرنامج سيقوم بتنفيذ العمليـة :

$$(150) + (151) \rightarrow (152)$$

لدينا إذن العلاقة التالية :

العنوان الفعلي = العنوان القاعدي + العنوان الموجود في التعليمة

نشير إلى أن عملية الجمع تتم ديناميكياً ، في لحظة تنفيذ كل تعليمة . يبدو من البديهي أن المبرمج لا يجب أن يُعدِّل مضمون المرصف القاعدي . العنوان النسبي الموجود في التعليمة يُدعى إزاحة (déplacement) .

المكنات IBM 360/370 تتمتع بـ 16 مرصفاً عاماً يمكن أن تُستعمل كمراصف قاعدية . يُحدَّد المرصف بالكامل بواسطة رقم المرصف المستعمل كمرصف قاعدي والعنوان النسبي . هكذا ، فإن حقل العنوان من تعليمات هذه المكنات سيحتوي على حيِّـز من أربع بتات حيث يتم تخزين رقم مرصف القاعدة .

الحسنات :

- يكتب المبرمج برنامجه بشكل مستقل عن الموقع الذي سيشغله في داخل الذاكرة .
- البرنامج ، أو مجموعة الحيزات والتعليمات ، هو قابل للتحويل والنقل . من الممكن نقله من حيِّـز من الذاكرة إلى حيِّـز آخر دون تعديل في العناوين المنقولة (المحوَّلة) . يكفي تعديل مضمون المرصف القاعدي .
- العنونة الأساسية وبشكل عام العنونة النسبية تسمح بعنونة مناطق كبيرة من الذاكرة بـدلاً من أن تحتوي التعليمـة ، على حقـل عنوان طـويل جـدّاً . نشير حـول هـذا الموضوع ، أنه لعنونة 2^n خلية من الذاكرة يلزمنا عدد n من البتات .

السيئات :

- كل تعديل في مرصف القاعدة خلال تنفيذ التعليمة يؤدي إلى نتائج غير متوقعة .

3.3 . العنونة المؤشرة (Indexed address)

يتعلَّـق ذلك بعملية حسابة العنوان بشكل شبيه بالعنونة القاعدية ولكن بهدف مختلف . يوجد مرصف يدعى مرصف التأشير أو مرصف الدليل (index register) ،

(1) العنوان القاعدي ليس بالضرورة عنوان زرع البرنامج .

تُخزَّن فيه قيمة معينة بواسطة المبرمج :
هكذا :

العنوان الفعلي = العنوان القاعدي + الإزاحة + مضمون المرصف المؤشر

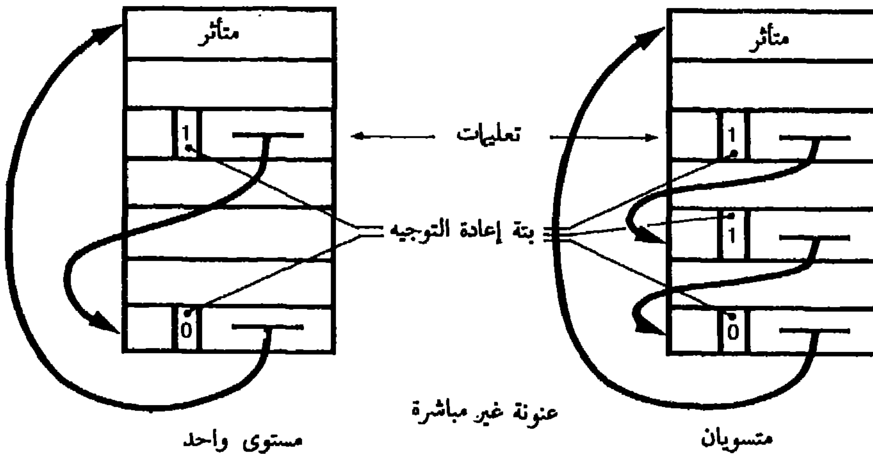
وعلى عكس مرصف القاعدة ، فإن مرصف التأشيرَ يُمكن أن يُعدَّل مضمونه بواسطة المبرمج . هذه الأوالية تسمح ، بواسطة عمليات الزيادة على مضمونه هذا،بأن نقوم بعمليات تكرارية ، وتشكيل حلقات (loop) من التعليمات ، وبالتالي بلوغ خلايا متتالية من الذاكرة . هذه هي التقنية المستعملة لبلوغ الجداول . التعليمات التي تعود إلى عناوين والتي يُمكن أن تحتمل عملية تأشير تتمتع بحقل إضافي خاص بالمرصف المؤشر جيث يستطيع المبرمج وضع رقم المرصف الذي يرغب باستعماله كدليل أو كمؤشر . (index)

4.3 . العنونة المباشرة

نتكلم عن العنونة المباشرة عندما نجد في التعليمة العنوان الفعلي للمتأثر . إنّـها إذن أوالية العنونة البسيطة والمطلقة .

5.3 . العنونة غير المباشرة

هذه التقنية في العنونة موجودة على أكثر المكنات،حقل العنوان من التعليمة لا يحتوي على عنوان المتأثر ولكن على كلمة تحتوي عنوان المتأثر . بعض المكنات تتمتع ، عتادياً ، بأداة خاصّـة لتغيير الإتجاه . في هذه الحالة ، يوجد بتة خاصة في التعليمة تشير إلى وجود أو عدم وجود إعادة تغيير في الاتجاه . إعادة التوجيه يمكن أن تتم في مستويات عديدة كما يبرهن لنا المثل التالي :



6.3 . العنونة التلقائية

هذا المصطلح الشائع هو سيء لأن هذه الطريقة لا تخص عنواناً معيّناً إنما تخص قيمة محدَّدة . المعلومة الموجودة في حقل التعليمة المُستعمل لكتابة العنوان ، لا تُمثِّل عنوان المتأثر ، إنما المتأثر نفسه (قيمة تستعملها التعليمة) .

تصفير المرصف يمكن أن يتم بطريقتين :

- بواسطة العنونة المباشرة يتم تصفير كلمة من الذاكرة بعنوان A ، وسنستعمل تعليمة لشحن المرصف بعنوان مباشر مع مضمون R:A → (A) ؛
- بواسطة العنونة التلقائية ، سيجري نقل القيمة صفر الموجودة في التعليمة على موقع العنوان إلى المرصف مع احتمال إزاحة البتة ذات الوزن الأكبر إلى اليسار إذا كان حجم حقل العنوان أصغر من حجم المرصف . العملية تتم بدون مساعدة أية خلية إضافية من الذاكرة . الحاسبات IBM 360/370 تتمتع بمجموعة من التعليمات ، تلك ذات الصيغة SI ، وتعمل بعنونة تلقائية .

4 هيكلية الحاسبات IBM 360 / 370

لن نقوم-هنا سوى بإيجاز المميزات الضرورية الواجب معرفتها للبرمجة . بعض النقاط يمكن أن تعتبر حاجزاً أمام القارىء المبتدىء ، وستتوضح له لاحقاً إلّا أنّنا وجدنا من المفيد تحديدها منذ الآن .

1.4 . الذاكرة

الذاكرة هي معنونة بالبايتات (فقرة 1.5.2) . وسعتها القصوى هي 16777216 بايتة (2^{24}) . تُرقَّـم البايتات على التوالي بدءاً من الصفر تجري التعليمات على سلاسل من البايتات ، نصف كلمات (عناوين مزدوجة) من بايتتين ، وعلى كلمات (عناوين تقبل القسمة على 4) من أربع بايتات وكلمات مزدوجة (عناوين مضاعفة لِـ 8) من ثمان بايتات . تُرقَّـم بتات الكلمات من اليسار إلى اليمين من 0 إلى 31 .

2.4 . المراصف

تستعمل مراصف التحكم بواسطة نظام التشغيل لإدارة الذاكرة . وهي مبلوغة بواسطة تعليمات مميَّـزة وخاصة ، لن نتكلَّـم عنها .

المراصف العامة وعددها 16 ومُرقَّـمة من 0 إلى 15 ، ويُمكن أن تُستعمل :

- كمراصف قاعدية (أساسية) (ما عدا المرصف 0) ، وتحتوي على عنوان مطلق من 24 بتة من اليمين .
- مراصف دليلية (مرصف مؤشر) (index register) (ما عدا المرصف رقم 0) .
- مرصف شحن (مركم) أو توسيع لمرصف الشحن يستعمل لإجراء العمليات على التمثيلات الداخلية للاعداد بفاصلة ثابتة أو عمليات منطقية . بعض العمليات تحتاج إلى وجود مرصفين « متلاحقين » (الضرب مثلاً) . نستعمل عندئذٍ مراصف عامة متتالية ، الأول يكون إلزامياً برقم مزدوج . سنُسمِّـي لاحقاً زوجاً من المراصف كهذا ، مرصفاً مزدوجاً . التعليمات التي تستعمل مرصفاً مزدوجاً لا تشير سوى إلى المرصف برقم مزدوج .

المراصف الأربعة المتحركة هي متخصّصة في الحسابات الجارية على الأعداد الممثلة بفاصلة متحركة . وتحمل الأرقام 0 ، 2 ، 4 ، 6 .

هذه المراصف هي بطول 64 بتة ويمكن أن تحتوي على عدد طويل بفاصلة متحركة أو عدد بطول قصير من نفس النوع . يشغل العدد القصير بفاصلة متحركة البتات ذات الأوزان العالية ، وتُهمل البتات الأخرى . والمراصف المستعملة لتخزين الأعداد الممثلة بفاصلة متحركة أو المراصف المتحركة يُمكن أن تزاوج (2—0 و6—4) بالنسبة للعمليات بالنسق الواسع (extended format) .

3.4 . الكلمة PSW (Program status word)

الكلمة PSW هي عبارة عن كلمة مزدوجة متعددة الأدوار . نجد فيها ، عند الانقطاع ، عنوان التعليمة التالية المطلوب تنفيذها . وتحتوي على نتائج عمليات المقارنة (كود ـ الشرط) ، ومعلومات عن بعض الحوادث (كود الانقطاع) . وتسمح بتنقيح حوادث الزيادة عن السعة (overflow) ، وتشير الى طريقة تشغيل الحاسب (الصيغة الرئيسية أو المُميّزة أو صيغة المسألة) .

معرفة الكلمة PSW المرتبطة بالبرنامج تترجم إذاً مفهومها الخاص بالتنفيذ . عند حدوث إنقطاع في البرنامج ، أي تعليق تنفيذه لمعالجة مسألة أكثر أولوية ، يتم تخزين الكلمة PSW الخاصة بالبرنامج المعلّق في الذاكرة ، وتدعى عند ذلك الكلمة « PSW القديمة » . الكلمة PSW الجديدة ، والمرتبطة بالبرنامج الجديد الذي يعالج الانقطاع ، يتم شحنها مما يؤدي إلى تنفيذ برنامج جديد . البرنامج المعلّق يمكن أن يُعاود تنفيذه بشرط ترميم أي إستعادة الكلمة PSW .

هناك طريقتان للتحكم موجودتان على المكنة 370 : الطريقة الأولى (Basic control mode)BC والطريقة EC (Extended control mode) .

وتختلف الطريقتان من حيث كون الترجمة الديناميكية للعنوان هي غير ممكنة سوى في الطريقة EC . وبكل طريقة في التحكم يرتبط نسق جديد للكلمة PSW . وتُميّزها بواسطة البتة رقم 12 .

1.3.4 ـ الكلمة PSW في الطريقة BC (bit 12= 0)

هذا هو نسق الكلمة PSW على المكنات IBM 360 .

قناع القنوات	I/O	E	مفتاح	C	M	P	W	كود الانقطاع
0	6	7		12				16 … 31

ILC	CC	قناع البرنامج	عنوان التعليمة
32	34	36	40 … 63

مخطط 1.4 . نسق الكلمة PSW في الصيغة BC

ـ الأقنعة . وهي مرتبطة بمختلف أسباب الانقطاعات . وجود البتة «0» في بتة القناع يمنع المعالجة المباشرة للحادثة . الإنقطاعات من نوع overflow (قناع البرنامج) يمكن أن تهمل ، وتوضع الأخرى في الانتظار حتى رفع أو زوال سبب المنع أو الإهمال . فقط بإمكان المبرمج بلوغ قناع البرنامج عندما يعمل الأخير في صيغة المسألة (bit 15=1) البتة رقم 15 تعادل 1 .

البتات من 0 إلى 6 تتعلّق بالإنقطاعات الآتية من القنوات . البتة 7 (E) ، الانقطاعات الخارجية ، البتة 13 (M) ، عمل المكنة السيء والبتات من 36 إلى 39 ، الانقطاعات الناتجة عن تجاوز في السعة ، البتة 36 مرتبطة بالفيض عن السعة (Overflow) أثناء إجراء العمليات الجبرية بفاصلة ثابتة ، والبتة 37 مُتعلّـقة بالنظام العشري والبتتان 38 و39 متعلّقتان بالحساب بفاصلة متحركة .

ـ مفتاح الحماية : هذا المؤشر (البتات من 8 إلى 11) ، وبالعلاقة مع المفتاح الموجود في الذاكرة ، يتيح أو يمنع بلوغ البرنامج إلى بعض المناطق من الذاكرة .

ـ البتة 12 (C) تشير إلى طريقة العمل في التحكم .C=0 تدل على طريقة العمل BC .

ـ البتة 14 (W) ، تساوي 1 عندما تكون الوحدة المركزية غير فعّـالة ، في حالة الإنتظار (Wait) .

ـ البتة 15 (P) تعادل 1 عندما تكون الوحدة المركزية في الصيغة مسألة ، والتعليمات المُميّـزة هي أيضاً ممنوعة . وتعادل هذه البتة صفراً في صيغة العمل (Supervisor) أي المشرف .

ـ كود الإنقطاع : عندما يحدث أي إنقطاع ، فإن الكلمة القديمة PSW للبرنامج المقطوع تُخزّن في الذاكرة ويُوجد فيها كود خاص يُعرِّف عن طبيعة الإنقطاع .

ـ ILC (البتتان 32 و33) (Instruction Length code) . عند حدوث إنقطاع نجد في هاتين البتتين طول آخر تعليمة جرى تفسيرها .

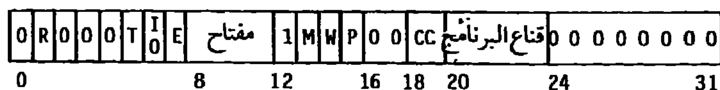
ـ CC (البتتان 34 و35) . عبارة عن الكود ـ الشرط الذي يعطي نتيجة المقارنة ، إشارة المتأثر بعد تعليمات عديدة . . .

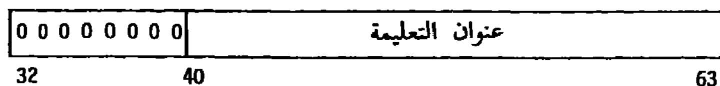
ـ عنوان التعليمة (البتات من 40 إلى 63) . هو عبارة عن عنوان التعليمة التالية المطلوب تنفيذها . تعرف PSW في لحظة الإنقطاع ، هذا الحقل يدل إذن على عنوان التعليمة حيث يجب أن يُعاوِد البرنامج عمله .

2.3.4 . الكلمة PSW في صيغة العمل EC (البتة 12=1) .

تختلف عن السابقة بواسطة إلغاء أقنعة القنوات ، وكود الإنقطاع والكود ILC . ويستبدل ذلك بواسطة قناع «R» يدعى «program event recording mask» وبتة T

تتعلق بطريقة نقل العناوين . دراسة هذه الإمكانيات تخرج عن إطار هذا الكتاب ، ولن نتكلّم عنها .





شكل 4.2 . النسق PSW في الصيغة EC

5 لغة الآلة

1.5 . نسق التعليمات الآلية

لقد أدّت بنا دراسة المكنة البسيطة إلى تعريف التعليمات الآلية بطول ثابت ، والمركّبة من كود للعملية ومن حقل للعنوان . تهتم العملية بمتأثر واحد ، بينما يكون المتأثر الثاني موجوداً في مرصف الشحن أو المركم (Accumulator) .

تتمتع المكنات IBM 360/370 بآوالية للعنونة أكثر تعقيداً ، تستعمل عدة مراصف وتتمتع بـ 16 مرصفاً عاماً يُمكن أن تُستعمل كمراصف شحن . نرى إذاً أن تعليمة بعنوان واحد ستكون مركبة من :

- كود للعملية (op. code) .
- رقم مرصف الشحن المعتمد في التعليمة .
- القسم عنوان الذي يتألف من :
  - رقم المرصف القاعدي ،
  - رقم المرصف الدليلي (المؤشر) إذا كان مستعملاً ،
  - قيمة الإزاحة .

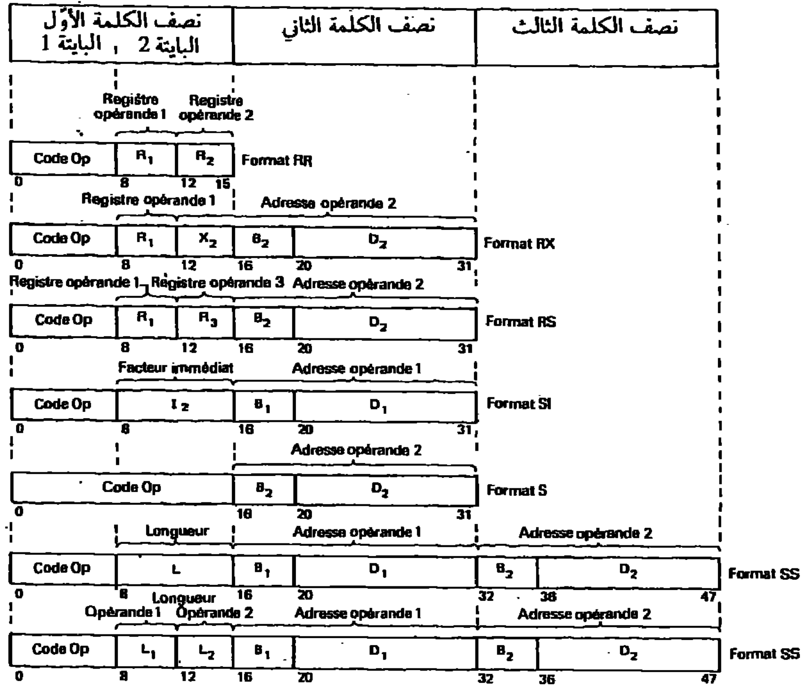
سيتم شرح تعليمات المكنات IBM 360/370 بواسطة ستة أشكال (نسق) مختلفة تتعلّق بطبيعة المتأثرات . التعليمات ذات النسق RR (Register to Register) لا تستعمل سوى مرصفين . التعليمات من نوع RX تعالج عدداً موجوداً في أحد المراصف وآخر على عنوان معين في الذاكرة وهذا العنوانُ يُمكن أن يكون دليلياً أو مؤشراً . النسق RS (Register and Storage) ، وSI (Storage Immédiat) ، و SS (Storage and Storage) ، وS لا تسمح بأي عملية تأشير.

الجدول التالي يُحدِّد نسق التعليمات المستعمل . الحقول R ، X ، B ، D تُمثّل على التوالي أرقام المراصف ، المراصف الدليلية ، مراصف القاعدة وقيمة الإزاحة . الحرف L يرمز إلى طول المتأثر ويُقاس بالبايتة في التعليمات بالنسق SS . الدليلان 1 و2

يربطان هذه المعلومات بالمتأثر الأول والثاني .

سنلاحظ إن البايتة الأولى تحتوي دائماً على كود العملية (ما عدا بالنسبة للنسق S الذي يستعمل 2 بايتة) ، إن نصفي الكلمة الثاني والثالث هما عبارة عن عناوين بشكل قاعدة وإزاحة . من المهم أن نتذكر أن التعليمات يجب أن تكون محصورة في نصف كلمات .

تتمتع التعليمة من نوع RX التي تستعمل عنواناً غير مؤشر بحيز X2 يعادل الصفر . والتعليمة التي تستعمل عناوين غير مرتكزة على قاعدة سيكون فيها الحيز B صفراً . وبالتالي : فإن المرصف 0 لا يُستعمل لا كدليل ولا كمرصف قاعدي .



جدول 1.5

كود العملية المبتدىء بـ Code opération commençant par	النسق Format	الطول بالبايتات Longueur en octets
00	RR	2
01	RX	4
10	RS، SI ، S أو RX	4
11	SS	6

جدول 2.5

وفي النهاية ، يُمكن أن نُذكر بأن البتتين رقم 1 و2 من كود العملية ترمزان إلى طول ونسق التعليمة . الجدول 2.5 يوجز لنا ذلك .

2.5 . فئات التعليمات

من الممكن تصنيف التعليمات الآلية ضمن ست فئات :

1 ـ تعليمات التبادل :
- من مرصف إلى مرصف .
- من الذاكرة إلى مرصف (شحن المرصف LOAD) .
- من مرصف إلى الذاكرة (STORE) .
- من الذاكرة إلى الذاكرة .
- شحن تلقائي لأحد المراصف .
- شحن تلقائي للذاكرة .

2 ـ التعليمات الحسابيـة :
- الجارية على أعداد بالنظام الثنائي البحت (فاصلة ثابتة) ،
- على أعداد بفاصلة متحركة ، بدقّـة بسيطة ، بدقة مزدوجة أو بنسق موسّـع ،
- على أعداد بالنظام العشري المُكثّـف ،
- عمليات المقارنة الحسابية .

3 ـ التعليمات المنطقيـة :
- التقاطع ، الاتحاد ، المكاملة . . .
- المقارنة المنطقية .

4 ـ تعليمات التحكم بتوالي التعليمات (تعديل مضمون عداد البرنامج PC) .
- تفريـع إلزامي . .
- تفريع مشروط .

5 ـ تعليمات الإدخال / الإخراج (Input / Output)

6 ـ تعليمات متفرّقة :
- تحويل النسق ، إختيار PSW ، الإزاحة . . .

هذه التعليمات تعالج كلمات ، نصف كلمات ، كلمات مزدوجة أو سلاسل من السمات . إضافة لذلك نجد عدة تعليمات للجمع حسب طول المتأثرات ، ومواقعها في الذاكرة أو في المراصف ، أو حسب تكويدها الداخلي . مجموع التعليمات يتجاوز إذاً 150 تعليمة .

3.5 . كتابة البرنامج بلغة الآلة

هدف هذا المثل هو الإعتياد على نسق التعليمات الآلية . نقترح جمع مضمون

كلمتين وخزن النتيجة في الذاكرة .

كما ذكرنا أعلاه ، فإن جميع العناوين تُحسب بالنسبة إلى قاعدة (أساس) . إلهم الأول للمبرمج هو في حفظ واحد من 15 مرصفاً عاماً كمرصف قاعدي . نختار مثلاً المرصف رقم 15 .

هكذا ، فإن جميع التعليمات التي تستعمل عناوين ستحتوي على «F» في الحقل المحفوظ للقاعدة .

كتابة البرنامج بلغة الآلة يتطلب إختياراً جيِّداً لعناوين وجود أو إدخال المعلومات في الذاكرة والمناطق المؤقتة لحفظ النتائج .

تسمح لنا أوالية العنونة القاعدية والإزاحة بعدم الاهتمام بالعنوان الفعلي للمعلومات في الذاكرة . نعتمد في تفكيرنا العناوين النسبية . لنفترض إذاً أن المتأثر الأوّل موجود على العنوان 0 والثاني في الكلمة التالية ، أي بدءاً من البايتة رقم 4 . لنختر الكلمة الثالثة لتخزين النتيجة . ولنفترض أيضاً ان المتأثر الأول يعادل 29 والثاني يعادل 3- . فلنجعل حيِّز النتيجة صفراً في البداية . وكي نستطيع تمثيل مضمون حيِّزات الذاكرة يجب علينا أيضاً تحديد طريقة التمثيل المُعتمدة للأعداد . ولنختر الأسهل ، صيغة الأعداد بفاصلة ثابتة . حيز المعطيات في برنامجنا هو إذاً ممثّل بالنظام السادس عشري على الطريقة التالية قبل تنفيذ البرنامج :

المتأثر الأول	المتأثر الثاني	المتأثر الثالث
0 0 0 0 0 0 1 D	F F F F F F F D	0 0 0 0 0 0 0 0
0	4	8 — 12

من الممكن تصوّر ثلاثة حلول مختلفة لكتابة برنامجنا :

الحلّ الأول

شحن (LOAD) المتأثر الأول في مرصف نعتبره لاحقاً مرصفاً للشحن من نوع Accumulator (يتم ذلك بواسطة تعليمة من نوع RX بين المرصف والذاكرة) ، جمع المتأثر الثاني إلى هذا المرصف (تعليمة RX) ، وخزن مضمون المرصف في حيِّز النتائج (تعليمة من نوع RX) .

لنختر المرصف 2 كمرصف للشحن (مركم) . كود عملية تعليمة الشحن (أنظر الملحق) هو 58 ، والتعليمة تكتب بالنظام السادس عشري :

- حيِّز كود العملية (COP) 58
- الحيِّز R1 2 (مرصف الشحن)

ـ حيِّز الدليل (index)　　　0 (بدون تأشير)
ـ مرصف القاعدة　　　F (المرصف 15.)
ـ الإزاحة　　　0

أي :

5 8	2	0	F	0 0 0
COP	R_1	X_2	B_2	D_2

تُمثَّل المعطيات بفاصلة ثابتة ، سنستعمل التعليمة بكود العملية 5A التي تؤمن جمع مضمون الخلية ذات العنوان $B_1 + X_2 + D_2$ إلى المرصف المذكور في الحيِّز R_1

أي :

5 A	2	0	F	0 0 4
COP	R_1	X_2	B_2	D_2

004 = إزاحة المتأثر الثاني بالنسبة إلى القاعدة .

وفي النهاية ، سنُخزِّن النتيجة (التعليمة STORE ، بالكود 50) في الكلمة الثالثة على العنوان 8 .

5 0	2	0	F	0 0 8
COP	R_1	X_2	B_2	D_2

بإمكاننا أن نفحص صورة البرنامج بعد خزنه في الذاكرة .

العناوين الموجودة هنا هي العناوين النسبية ولا تتأثر بالعنوان الفعلي لموقع تخزين البرنامج . عنوان الاطلاق في التنفيذ ، أي عنوان أول تعليمة للتنفيذ ، هو عنوان القاعدة + C .

	0	0 0 0 0 0 0 1 D
	4	F F F F F F F D
	8	0 0 0 0 0 0 0 0
عنوان الاطلاق في التنفيذ	C	5 8 2 0 F 0 0 0
	10	5 A 2 0 F 0 0 4
	14	5 0 2 0 F 0 0 8
	18	

الحلّ الثاني :

إشحن المتأثرين الأول والثاني في المراصف ، وقم بعملية جمع لمضمون مرصف مع المرصف الآخر ومن ثم خزِّن النتيجة . نستعمل المراصف 2 و3 كمراصف للعمل والمرصف رقم 15 كمرصف قاعدي . والبرنامج هو التالي :

0	0 0 0 0 0 0 1 D	
4	F F F F F F F D	
8	0 0 0 0 0 0 0 0	
C	5 8 2 0 F 0 0 0	شحن المتأثر الأول في R2
10	5 8 3 0 F 0 0 4	شحن المتأثر الثاني في R3
14	1 A 2 3	جمع في R2
16	5 0 2 0 F 0 0 8	خزن النتيجة

هذا الحلّ يحتاج إلى تعليمة إضافية . سنلاحظ وجود تعليمة من نوع RR بطول 2 بايتة .

الحل الثالث :

الحلّ الثالث كان سيقوم على إجراء الحساب مباشرة في الذاكرة دون استعمال المراصف . وسيحتاج إلى وجود تعليمة بثلاثة عناوين (المتأثر الأول ، المتأثر الثاني والنتيجة) . إلّا أنّ هذا النوع من التعليمات هو غير موجود هنا .

خلاصة

نلاحظ ، في الأمثلة المذكورة ، أنّ حيّز المؤشّر (index zone) غير المستعمل هو مصفّر تماماً كما ذكرنا في الفقرة 1.5 .

إنّ البرمجة بلغة الآلة تبدو معقّدة ودقيقة رغم بساطة المثل وعدم إتمامه . لهذا السبب لا نستعمل هذا النوع من البرمجة ونفضّل عليه مرونة لغة المؤوّل (الأسمبلر) .

6 . لغة المؤول
ASSEMBLER

المثل البسيط الذي جري عرضه في الفصل السابق أثبت لنا جميع صعوبات البرمجة بلغة الآلة مع أنه جرى تبسيط كبير لعملنا باستعمال النظام السادس عشري بدلاً من النظام الثنائي .

في لغة الآلة ، فإن أكواد العمليات والعناوين هي رقمية . وكل تعديل في موقع المعطيات يؤدي إلى تعديل العناوين في التعليمات المتعلقة بها .

هذه الصعوبات أدت بالمصممين الى تعريف لغة ، تُدعى المؤول (assembler) ، قريبة من لغة الآلة ولكنها سهلة الإستعمال مما يجعل ترتيبها في مصاف اللغات المتطورة .

1.6 . مميزات لغات التأويل

1 ـ تتميز التعليمات بلغة المؤول بكود عمليات تذكيري . مثلاً : تعليمة شحن المرصف من خلال مرصف آخر تتمتع بكود رمزي هو LR (LOAD TYPE RR) ، وتمتاز تعليمات الجمع بكود رمزي يبدأ بالحرف A . . .

2 ـ بإمكان المبرمج أن يقوم بتحديد عناوين بواسطة أسماء رمزية ويقوم برنامج ترجمة المؤول إلى لغة الآلة بربط القيمة الرقمية المناسبة بهذه الأسماء .

3 ـ تتمتع لغة المؤوّل ليس فقط بمجموعة التعليمات الآلية التي تتضمنها لغة الآلة ، ولكن ببعض التعليمات الخاصة الآلية التي تدعى (التوجيهات) (أو أشباه التعليمات Pseudo-Instruction) وببعض الماكرو تعليمات (macro-instructions) .

2.6 . تعريفات

تدعى تعليمة ـ آلية كل تعليمة مكتوبة بلغة المؤول ومترجمة إلى تعليمة واحدة فقط بلغة الآلة . يتناسب كود رقمي مع كود ـ العملية التذكيري . مثلاً ، عملية نسخ المرصف 12 في المرصف 3 ، تكتب بلغة المؤول على الشكل التالي :

LR 3, 12 (LR = Load type RR)

وتُترجم إلى لغة الآلة بواسطة :

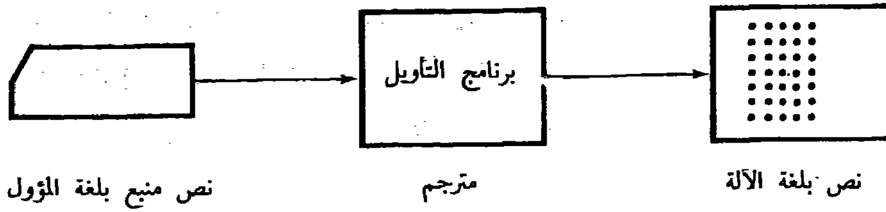
1 8	3	C
COP	R_1	R_2

يُدعى أمر من نوع توجيه directive كل طلب إلى المؤول ، لا يُولِّـد أبداً تعليمة آلية ولكن يُقدم توجيهات للتأويل والتجميع . يوجد نوعان من التوجيهات : تلك التي لا تؤدي إلى أية عملية حجز للذاكرة وتلك المستعملة لحفظ موقع من الذاكرة أو تعريف الثوابت المفيدة للمسألة . هكذا ، فالتعليمة USING*,15 تعني إن المرصف 15 سيُعتبر أولاً كمرصف قاعدي ، مما سيسمح بعدم ذكر القاعدة (Base) في التعليمات التالية . هذا التوجيه لا يشغل مكاناً من الذاكرة في الكود المولِّـد ، وليس هو سوى إشارة إلى برنامج التأويل والتجميع . أن نكتب 'DC X'F0F0 يعني أن نطلب إلى المؤول حجز بايتتين من أجل تخزين الثابتة المحدَّدة بالنظام السادس عشري بواسطة F0 F0 . لا يـوجـد تـوليـد لتعليمـة ولكـن فقط حفظ لمكـانٍ من الـذاكـرة . من الممكن تشبيـه هذه التوجيهات بتعليمات التصريح في اللغات المتطورة . أن نكتب بلغة فورتران الأمر dimension TAB (100) يعني أن نطلب من المصرِّف (Compiler) حفظ المكان من الذاكرة اللازم لاستيعاب الجدول (100) TAB .

سنسمّي ماكرو ـ تعليمة (MACRO-INSTRUCTION) كل طلب إلى البرنامج المؤوّل assembler باستبدال سلسلة معرَّفة مسبقاً من التعليمات تدعى ماكرو ـ تعريف . الماكرو تعريف هو إذاً عبارة عن مجموعة من التعليمات ينسخها البرنامج assembler مكان كل ماكرو ـ تعليمة . يقدم النظام مجموعة من الماكرو ـ تعريفات تدعى نموذجية (ستاندارد) تُسهِّـل على المبرمج القيام ببعض العمليات المعقدة ، كعمليات الإدخال ـ الإخراج . كما باستطاعة المبرمج أن يقوم بتعريف نظام خاص به من الماكرو ـ تعريفات .

3.6 عملية التأويل

الإسم «assembler» يعني في نفس الوقت اللغة والبرنامج الذي يقوم بترجمة النص إلى لغة ـ الآلة . سنقوم هنا بتناول مرحلة الترجمة بصورة موجزة . يبدو المؤول وكأنه عبارة عن مصرِّف أو كأنه عبارة عن برنامج لترجمة النص المكتوب بلغة منبع إلى نص مستهدف يتألف من تعليمات ـ آلية . تدعى عملية الترجمة تأويلاً «assembling» .



1.3.6 . عداد المواقع

يجب على المؤول ، ومن خلال نص منبع ، أن ينتج نصاً ثنائياً يكون مع بعض التحويلات عبارة عن صورة البرنامج المطلوب تنفيذه . لتخصيص عناوين متتالية للتعليمات ، يستعمل المؤول عداداً للمواقع نرمز إليه بواسطة CE . في بداية عملية التأويل فإن CE يهيىء ، مثلاً يُصفّـر . وخلال ترجمة التعليمات فإنه يزيد من قيمته حسب طول التعليمات المترجمة . وعندما يلتقي توجيهاً من نوع حجز لموقع أو منطقة من الذاكرة ، فإن مضمون CE يزداد حسب طول المنطقة المحجوزة . كل توجيه من نوع إشارة إلى برنامج التأويل لا يؤدي إلى زيادة في مضمون CE لعدم توليد أية تعليمة آلية . التعليمات ذات النسق RR تؤدي إلى زيادة مضمون CE 2 بايتة ، أما تلك التي تتمتع بنسق RX ، RS وSI فتؤدي إلى زيادة أربع بايتات إلى مضمون CE ، أمّا تلك ذات النسق SS فتؤدي إلى زيادة 6 إلى مضمونه . وكل توجيه لحجز كلمتين من الذاكرة يؤدي إلى زيادة مضمونه 8 بايتات .

في المثل التالي ، STARTO هي عبارة عن توجيه يؤدي إلى تهيئة CE وتصفيره . لا يحدث أي توليد لتعليمات جديدة وبالتالي فإن CE يبقى صفراً . STM 14, 12, 12(13) هي عبارة عن تعليمة من نوع RS تؤدي إلى زيادة 4 إلى مضمون CE . والتوجيه DS 1F يؤدي إلى حفظ كلمة من الذاكرة يُرمز إليها بواسطة ALPHA . وCE تزداد قيمته 4 بايتات . التعليمة LR 0,1 بالنسق RR تجعل مضمون CE يزداد 2 .

CE بالنظام السادس عشري	العنوان الرمزي	كود العملية	منطقة المتأثرات	ملاحظات
0		START	0	تصفير CE
0		STM	14,12,12(13)	تعليمة من نوع RS
4		---	------------	
--		---	------------	
20	ALPHA	DS	1F	حجز كلمة
24		---	------------	
--		---	------------	
48	DEBUT	LR	0,1	تعليمة من نوع RR
4A		---	------------	

وبالاختصار ، فإن عداد المواقع هو عبارة عن كلمة ـ ذاكرة يُخزِّن فيها المؤول :

قبل تأويل التعليمة ، عنوان بداية التعليمة (المتعلّق بتهيئة CE) ،

ـ بعد التأويل ، عنوان الخلية الأولى المتوفّرة .

من الممكن أن نلاحظ إن قيمة CE تعادل قيمة مضمون عداد البرنامج عند التنفيذ .

2.3.6 . العنونة الرمزيّة والمرجعيات المطلقة

لقد ذكرنا سابقاً أنّ أحد أهم مميزات وخصائص المؤول تكمن في إمكان تسمية العناوين والقيم بواسطة رموز . يمكن أن يكون الرمز عبارة عن إسم منطقة من الذاكرة . في الجدول السابق ، فإنّ ALPHA وDEBUT هما عبارة عن عنوانين رمزيين نستطيع بلوغهما والعودة إليهما . سيكون بإمكان المبرمج أن يراجع مناطق من الذاكرة تبعاً لهذين العنوانين بواسطة تعابير من نوع DEBUT − 2, ALPHA + 8 .

يُستعمل الرمز * لتسمية القيمة التي يأخذها CE في لحظة التأويل ، أي عنوان البايتة اليسرى من التعليمة الموجودة في طور التأويل . من الممكن أن نعود أيضاً بواسطة 2− * إلى العنوان الجاري ناقص 2 بايتة .

سنلاحظ أيضاً أنّـه لا يمكن لقيمتين مختلفتين لمضمون CE أن تحمـلا نفس الإسم . إذ نكون عندئذٍ في حالة التعريف المزدوج .

يسمـح المؤول أيضاً ببلوغ قيم مطلقة بواسطة رموز ، أي رموز غير متغيِّـرة عند ترجمة البرنامج . تُكتب عملية نسخ المرصف 1 في المرصف 0 مثلاً : LR 0,1 .

يمكننا أيضاً أن نكتب ، بشكل أوضح LR R0, R1 بشرط تحديد كون R0 وR1 عبارة عن رمزين مطلقين يعادلان القيمتين 0 و1 .

وفي النتيجة ، فإن المؤول سيربط بكل رمز قيمة تدعى قيمة ـ خاصية ، وهذه القيمة سيتم ترجمتها أو عدمه حسب الحالة .

3.3.6 . جدول الرموز

عند العمل ، وفي كل مرّة يلتقي المؤول رمزاً معيناً في منطقة الوسم (Label) يقوم بتخصيص خاصيات له :

ـ خاصية ـ قيمة تعادل قيمة CE في هذا الموقع .

ـ خاصية ـ طول تعادل البعد (الحجم) بالبايتات للمنطقة المعينة .

يمكن أن يقوم المؤول إذاً ببناء جدول من الرموز على الشكل التالي :

وسم رمزي	خاصية ـ قيمة	خاصية ـ طول
ALPHA	20	4
BÉTA	...	...
DÉBUT	48	2
...	...	...

عندما يلتقي رمزاً معيناً في قسم العنوان من التعليمة ، يقوم المؤول باستشارة هذا الجدول . فإذا كان هذا الرمز موجوداً فيه معنى ذلك أنّ الرمز محدّد مسبقاً، وإلا فذلك يعني مرجعاً إلى الأمام ، أي إنه لم يلتقِ الرمز حتى الآن في منطقة الوسم ولكنه سيكون لاحقاً (إلّا إذا كان يتعلق ذلك برمز خارجي ، أنظر الفصلين 20 و21) .

4.3.6 . تأويل التعليمة

يتعلّق ذلك باختيار كيفية ترجمة التعليمة بواسطة المؤول وبالأخص كيف يقوم بتحويل العنوان الرمزي الى عنوان قاعدي ، مؤشر وإزاحة . سنقوم بتحليل ذلك من خلال مثل معين .

لنفترض التعليمة التالية :

L 12, ALPHA

العامل الأوّل: 12 — العامل الثاني: ALPHA

إنّها تعليمة من نوع RX وبكود عملية 58 (أنظر الملحق) حيث معناها هو « شحن مضمون الخلية ذات العنوان ALPHA في المرصف رقم 12 » . يقوم عمل المؤول على تعبئة مختلف حقول التعليمة بالنسق RX ، أو :

5	8	C	0		
COP		R_1	X_2	B_2	D_2

منطقة العنوان (X_2 B_2 D_2)

فلنلاحظ منذ الآن إن منطقة الدليل هي صفر ، لأنه لم يذكر أي مرصف مؤشر أو دليل في العامل الثاني من التعليمة (الحقل الثاني منها) . ولتكملة حيز العنوان ـ يجب :

- معرفة المرصف المستعمل كقاعدة ،
- معرفة إزاحة العنوان ALPHA بالنسبة للعنوان القاعدي .

ونشير إلى أن العنوان القاعدي لا يختلط بالضرورة مع عنوان وجود البرنامج في الذاكرة .

سنقوم بافتراض في المثل إن ALPHA تناسب القيمة 1C للعداد CE ، وإن المرصف 15 هو مرصف القاعدة وإن العنوان القاعدي يناسب القيمة C للعداد CE . إزاحة ALPHA بالنسبة إلى القاعدة هي إذاً C-C1 أي 10 . التعليمة الآلية المؤولة ستكون إذاً :

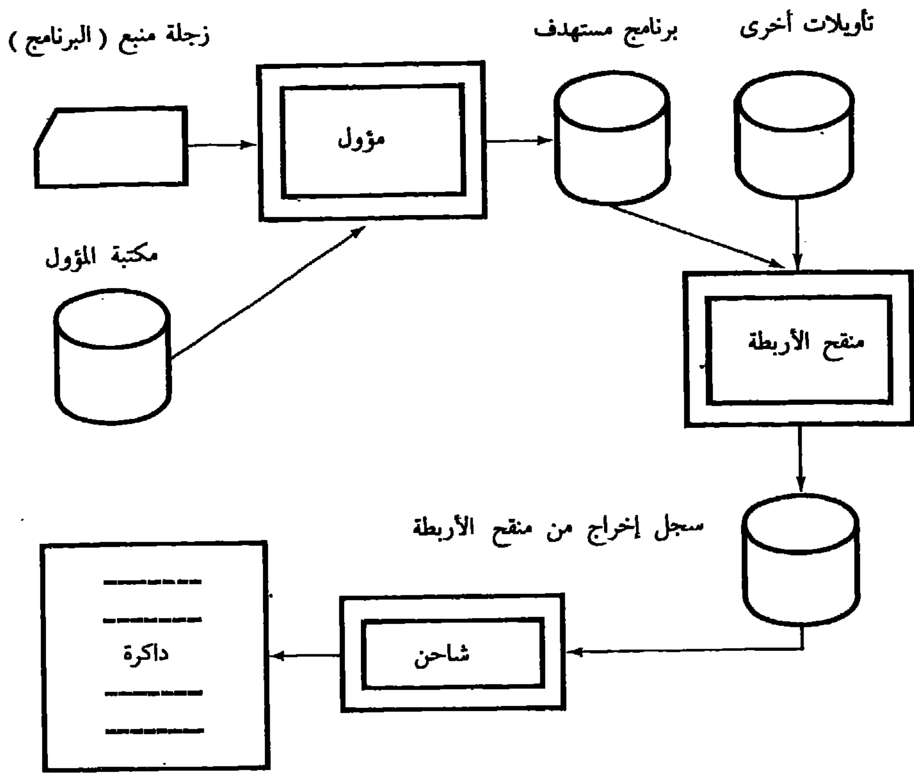
5	8	C	0	F	0	1	0

4.6 . مراحل تنفيذ البرنامج

إنّ تنفيذ البرنامج المكتوب بلغة المؤول ، كما بالنسبة للبرنامج المكتوب بإحدى اللغات المتطورة ، يتطلب عدة مراحل . المرحلة الأولى هي مرحلة التأويل والتجميع التي تكلمنا عنها . يُترجم النص الأولي الى لغة الآلة ويُنسخ في سجل على الاسطوانة المغناطيسية . المرحلة الثانية ، التي يمكن أن تكون اختيارية للبرامج البسيطة ، هي تنقيح الأربطة (link editor) . وتؤدي إلى إجراء بعض الوصلات بين مختلف الزجل المؤولة بشكل منفصل أو التي تشكّل جزءاً من مكتبة البرامج . منقّح الأربطة يُشكّل زجلة واحدة مستهدفة ، يمكن أن تتمتع بهيكلية تغطية ، من خلال مختلف عمليات التأويل . المرحلة التالية تقوم على شحن الزجلة في الذاكرة ، أي إعطائها عنواناً فعلياً لخزنها . وفي هذه الحالة تكون العناوين القاعدية متجمّدة ، وبعض المعلومات المُتعلقة بالعناوين المطلقة يجب أن تُحسب من جديد . يكفي إذاً أن نقوم بتخزين عنوان أول تعليمة للتنفيذ في عداد البرنامج CO (الكلمة الثانية من PSW) للبدء بمرحلة التنفيذ .

سنسمّي نقطة الشحن أو عنوان الخزن ، عنوان بداية المنطقة المُخصّصة للبرنامج . سيُدعى عنوان الإطلاق عنوان أول تعليمة للتنفيذ من البرنامج . نقاط الدخول الى البرنامج هي عناوين ، التعليمات أو المعطيات ، من الممكن بلوغها من خارج البرنامج . تتصل نقاط الدخول هذه بمنقّح الأربطة الذي يمكن أن يقوم بإجراء وصلات بين مختلف الزجل (modules) . عنوان الإطلاق هو نقطة دخول .

بدون إعطاء جميع الإمكانيات فإن المخطط 1.6 يعرض مختلف المراحل الواجب أن يتبعها البرنامج كي يجري تنفيذه .


زجلة منبع (البرنامج)
مؤول
برنامج مستهدف
تأويلات أخرى
مكتبة المؤول
منقح الأربطة
سجل إخراج من منقح الأربطة
داكرة
شاحن

مخطط 1.6

القسم الثاني

المؤول 370/360

7 . العناصر الأساسية

1.7 . عموميات وتقديم البرنامج

1 ـ مجموعة السمات :

يستعمل المؤول السمات الأبجعددية A ، B ، ..Z،@، $ ، والأرقام 0 ، 1 ، 2 . . . 9 ، والسمات الخاصة : + - * / = ( ) . ، ' ، & والقسمة البياض (blanck) .

2 ـ ورقة البرنامج

المنطقة المحجوزة للمؤول تمتد من العامود 1 إلى العامود 71 . المنطقة 73 إلى 80 لا تُفسّر من جانب المؤول وتُستعمل لتعريف التعليمات . العامود 72 يُستعمل عندما ترغب إحدى التعليمات بالمتابعة على السطر التالي . تقسّم منطقة التعليمة (1 إلى 71) إلى أربعة أقسام :

منطقة الرموز : وتُستعمل لاجراء تخصيص رمزي للتعليمة (وسم) أو إلى معطى (إسم المعطى) .

الاسم المُخصّص :

- يبدأ بالعامود 1 بواسطة سمة أبجدية .
- يحتوي على أكثر من 8 سمات أبجعددية .
- لا يحتوي على فراغ أو سمات خاصة .

الرموز التي تظهر في منطقة المتأثرات تخضع لنفس القواعد :

أمثلة :

صالح	غير صالح	
A1234567	RESULTATS	(9 سمات)
ZONE	TAB 1	(فراغ)
@123	1ABC	(تبدأ برقم)
###	BC–1	(تحتوي على سمة خاصة)
$ABC		

منطقة العملية : وتستعمل لتحديد كود ـ العملية الخاص بالتعليمة . هذا الحيِّز يبدأ في أي مكان ، إنطلاقاً من العامود رقم 2 . إلاّ أنّه يجب أن ينفصل الرمز عن كود العملية بواسطة فراغ واحد على الأقل .

منطقة العوامل (العناوين) : وتحتوي على العناوين أو على المتأثرات . تبدأ هذه المنطقة من أي عامود على يمين كود ـ العملية وتنفصل عنه بواسطة فراغ واحد على الأقل . ويمكن أن تحتوي هذه المنطقة على العناوين ، ولا يُمكن أن تحتوي على فراغات وكل عنوان ينفصل عن الآخر بواسطة فاصلة .

منطقة الملاحظيات : وتبدأ من يمين أول فراغ يتلو منطقة العوامل وتمتد حتى 71 عاموداً . يمكن إعتبار السطر بكامله كملاحظية فيما لو بدأ هذا السطر بنجمة (*) على العامود الأول .

سطر التكملة : كل سمة عدا الفراغ في العامود 72 تشير إلى أن التعليمة الجارية لم تنته وستتابع على السطر التالي . يفترض المؤول أن السطر التالي يبدأ بالعامود رقم 16 ، وبالنتيجة فإنّ التعليمة ستتابع بدءاً من العامود رقم 16 . يسمح بسطرين فقط لتكملة التعليمة .

الحصر العادي : من المفيد حصر مختلف هذه المناطق انطلاقاً من الأعمدة 1 ، 10 ، 16 و40 . ونشير إلى أن الحيز المُفسَّر بواسطة المؤول يمتد إلزامياً من 1 إلى 71 وإن الأسطر التابعة تبدأ من العامود رقم 16 . هذه القيم هي قابلة للتعديل بواسطة الأمر ICTL

منطقة الرموز	منطقة كود العملية	منطقة العوامل	منطقة الملاحظية		منطقة المعرِّف
1	10	16	40	72	80
			عامود تابع		
ALPHA	DC	C'ABCD'	colonne suite		
	LR	1,2			
* CETTE LIGNE EST UN COMMENTAIRE			(سطر ملاحظية)		
BETA	DC	C'TEXTE	.... SE CONTINUAN	*	(نص يتبع
		T SUR LA LIGNE SUIVANTE'	على السطر التالي)		

جدول 1.7

2.7 . عناصر لغة المؤول

لقد لاحظنا حتى الآن إن المؤول يسمح لنا باستعمال رموز معينة لتسمية العناوين أو القيم . وعملياً فإن لغة المؤول تسمح لنا :

- باستعمال كتابات مثل 'B'1011 ، 'X 'A10C والتي ستعامل وكأنها قيم باللغة الثنائية ، أو السادس عشرية . . . وهي ستكون عبارة عن القيم المعرَّفة أوتوماتيكياً .
- بلوغ الطول المتعلِّق بأحد الرموز . لو إفترضنا إن «BIDON» هو وسم تعليمة ، أو بشكل عام ، أكثر اسم حيِّز معينّ ، فإنّ L'BIDON سيحدِّد طول التعليمة أو المنطقة . ويتعلَّق ذلك بالخاصية ـ طول ؛
- إستعمال الأحرف كمتأثرات في التعليمات ؛
- خلط كل هذه الإمكانيات لنحصل على تعابير ستكون معادلة لعناوين قابلة للنقل إلى قيم مطلقة .

من الملائم إذاً تحديد القواعد النحوية التي تسمح باستعمال هذه الإمكانيات

1.2.7 . قيم المُعرَّفات الأوتوماتيكية (Auto-definition)

قيمة المعرَّف الأوتوماتيكي هي واحد من أشكال الكتابة ، معروف من قبل المؤول ، يسمح بتحديد القيمة .

مثلاً :

X'B' ، 'B'1011 و 11 هي عبارة عن ثلاث كتابات مختلفة تسمح بتحديد القيمة 11 (عشري) الممثلة في المكنة بواسطة تشكيلة البتات 1011 . هذا الشكل في الكتابة هو مسموح ، مع بعض التحديدات ، بداخل حيِّز العوامل (منطقة العنوان) من التعليمة .

هناك أربعة أنواع من المعرَّفات الأوتوماتيكية المقبولة :

- الثنائي : 'B'1001101 وعلى الأكثر 32 رقماً ثنائياً تحت إشراف النظام OS و24 بالنظام DOS) .
- السادس عشري : 'X '1A3BC ،
- العشري : 125 (حدٌّ أقصى 10 أرقام عشرية) .
- نوع السمات : 'C'A ، '''C (سمة أبوستروف أو الفاصلة العليا) ، 'C'ABCD ، 'C'AB . يجب أن نحصل كحدٍ أقصى على أربع سمات بالنظام OS وثلاث بالنظام DOS .

وبشكل عام ، فإن قيمة التعريف الأوتوماتيكي يجب أن تتم على 24 بتة بإشراف النظام DOS وعلى 32 بتة كحدٌّ أقصى بإشراف النظام OS . سنجد أمثلة على طرق إستعمالها في الفقرة 3.7 المتعلقة بالتعابير .

2.2.7 . المتأثرات الحرفية

هي عبارة عن قيم مستعملة كمتأثرات في حيّـز عوامل التعليمات .

لشحن القيمة 125 في المرصف 3 يمكن للمبرمج أن يختار أحد حلّين :

1 ـ حجز حيز من الذاكرة ، يدعى ALPHA مثلاً ، ويُعرّف عنه وكأنه يحتوي على القيمة 125 ، وبعد ذلك يُشحن ALPHA في المرصف 3 بواسطة التعليمة :

L3, ALPHA;

2 ـ كتابة التعليمة : L 3, = F'125' ، وسيهتم المؤول بحجز الخلية من الذاكرة التي تحتوي على 125 في منطقة تدعى POOL (حوض) .

في المثل المذكور لاحقاً ، فإن القارىء سيتحقّق :

- من أن المؤول سيضع عنوان المتأثر الحرفي بشكل قاعدة وإزاحة داخل كود التعليمة المولّـد عنه ،

- من أنّ إستعمالين مختلفين لنفس المتأثر الحرفي لن يؤدّيا سوى إلى حجز واحد في الذاكرة ،

- من أنّ المتأثر الحرفي هو شبيه برمز قابل للترجمة .

إنّ استعمال المتأثر الحرفي ، إن لم يحمل أي شيء جديد ، فإنه يُقدم لنا فائدة بالنسبة لوضوح كتابة التعليمة .

قواعد الكتابة

يُحدَّد المتأثر الحرفي وكأنه متأثـر عادي في توجيه DC مسبوق بالإشارة « = » . أما القواعد المتعلقة بمتأثرات التوجيه DC فإنها ستوضح لاحقاً .

- لا يمكن أن يُستعمل المتأثر الحرْفي كمُعامِل في التعبير (فقرة 3.7) الرقمي أو غير الرقمي .

- من البديهي ، لأن المتأثر الحرفي يُستعمل « كمعطى للإدخال » في التعليمة ، أن لا يظهر في الحقل المُستقبِل من التعليمة . سيكون من المتنافر أن نكتب: ST 3,=F'125' (ST = خزّن مضمون المرصف في الذاكرة) .

```
LOC    OBJECT CODE       ADDR1 ADDR2  STMT    SOURCE STATEMENT
000000                                  1           CSECT
                                        2           EXTRN SP1
                           00000        3           USING *,15
000000 5810 F018         00018          4           L     1,=F'0'
000004 5820 F01C         0001C          5           L     2,=C'ABCD'
000008 5820 F018         00018          6           L     2,=F'0'
00000C 5820 F020         00020          7           L     2,=V(SP)
000010 5830 F01C         0001C          8           L     3,=C'ABCD'
000014 5810 F024         00024          9           L     1,=A(SP1)
                                       10           END
000018 00000000                        11                 =F'0'
00001C C1C2C3C4                        12                 =C'ABCD'
000020 00000000                        13                 =V(SP)
000024 00000000                        14                 =A(SP1)
```

3.2.7 . الخاصية ـ طول

وتسمح ببلوغ الطول المرتبط بالرمز . ويُكتب :

مثلاً : L ' symbolic name   L ' إسم رمزي

L'ZONE   L'SUITE   L'*

- إذا كان الرمز هو إسم الحيز ، فهو يأخذ كقيمة طول الحيِّز مُقاساً بالبايتة .
- إذا كان الرمز هو إسم التعليمة ، فهو يأخذ واحدة من القيم 2 ، 4 أو 6 حسب نسق التعليمة .
- إذا كان الرمز هو « * » ، فهو يأخذ كقيمة طول التعليمة التي يظهر فيها .

بالنسبة للتوجيهين DC وDS . فإنّ الخاصية ـ طول لا تتأثّر بوجود عامل الإزدواجية . سنلاحظ أنه بالنسبة للتوجيه EQU فإن قيمة الخاصية ـ طول هي قيمة المتأثر الأيسر .

الأمثلة التالية ، وللفهم الكامل ، تتطلب بأن نكون أكثر تقدماً في هذه الدراسة . إلاّ أنّنا نعرضها هنا :

الرمز	كود ـ العملية	عوامل	خاصية	قيمة
ZONE1	DS	CL80	L'ZONE1	80
ZONE2	DS	CL200	L'ZONE2	200
CARAC	DC	C'ABCDE'	L'CARAC	5
ABSOL1	EQU	ZONE2-ZONE1	L'ABSOL1	200
ABSOL2	EQU	25	L'ABSOL2	1
INSTR1	LR	0,1	L'INSTR1,	2
			L'*	2
INSTR2	MVC	ZONE2(L'*),ZONE1	L'INSTR2	6
			L'*	6
	MVC	ZONE2(L'ZONE2-10),ZONE1	L'ZONE2	200
ALPHA	DC	6F'0'	L'ALPHA	4

3.7 التعابير

تعريف :

التعبير هو تركيب من الرموز ، وقيم التعريف ـ الأوتوماتيكي وخاصيات ـ الطول في منطقة المتأثرات من التعليمة .

الاستعمال :

تستعمل التعابير لتحديد :

- العنوان ،
- الطول الواضح ،
- المعدِّل ،
- عامِل التكرار .
- المتأثر .

فئات التعابير

التعابير هي بسيطة أو مركبة ، مطلقة أو قابلة للترجمة التغير البسيط هو الرمز الوحيد أو الرمز « * » (قيمة عدّاد المواقع عند تأويل التعليمة ، فقرة 1.3.6) .

التعبير المركّب هو مجموعة من عدة تعابير بسيطة مرتبطة بمؤثرات من نوع + ، - ، * ،[1] أو / ، التي تمثّل على التوالي الجمع ، الطرح ، الضرب والقسمة .

أمثلة :

ALPHA+2	*+3 * = CE
ALPHA-BETA	*-2 * = CE
3*DELTA.	A*3 * = مؤثر
(ALPHA-BETA)/2	**2 تعبير غير صالح
ALPHA+X'1A'	*3- تعبير غير صالح
TAB+L'LIGNE	

قواعد الإنشاء

التعبير المركب :

- لا يمكن أن يبدأ بمؤثر ،
- لا يمكن أن يحتوي على مؤثرين ثنائيين متتاليين ،

(1) يجب عدم الخلط بين المؤثر * والرمز الذي يمثل عداد المواقع .

ـ لا يمكن أن يحتوي على نجمتين ،
ـ لا يمكن أن يحتوي على تعبيرين بسيطين يتتابعان بدون مؤثر بينهما ،
ـ لا يمكن أن يحتوي على متأثر حرفي .

النظام OS يسمح باستعمال 19 مؤثراً أحادياً وثنائياً و6 مستويات من الأهلّة . بينما النظام DOS لا يسمح سوى بـ 15 مؤثراً و5 مستويات .

تقييم التعابير

يقوم المؤول بتخصيص قيمة رقمية لكل تعبير بسيط وبعد ذلك يُقيِّم من اليسار إلى اليمين التعبير حسب أولوية خاصة للضرب وللقسمة بالنسبة للجمع والطرح . C * A+B تُقيِّم وكأنها (B*C)+A وليس كأنها C* (A+B) . النتيجة الحسابية تصبح قيمة التعبير ، والمؤول يُقيِّم بشكل طبيعي في المكان الأول المؤثرات الأحادية وداخل الأهلّة . القسمة على صفر هي صحيحة وتعطي نتيجة صفر .

تعابير مطلقة ، تعابير منقولة

التعبير المنقول هو تعبير حيث القيمة تتغيَّر مقدار n إذا كان البرنامج منقولاً إلى n بايتة

التعبير المطلق هو التعبير الذي لا تتغيَّر قيمته عند النقل .

أمثلة :

لنفترض إن ALPHA وBETA هي رموز منقولة وإن VAL1 وVAL2 هي رموز مطلقة :

تعابير مطلقة	تعابير منقولة
VAL1+B'101'	ALPHA+3
ALPHA-BETA	BETA+L'ZONE
VAL1+VAL2	BETA+VAL1

التعبير سيكون مطلقاً إذا كان يحتوي على :
ـ رموز مطلقة ، قيم تعريفات أوتوماتيكية ، خاصيات ـ طول ،
ـ رموز منقولة يظهر كل اثنين منها على حدة وتؤدي إلى تصفير فاعلية النقل .

سنلاحظ إنه إذا كان T1 وT2 تعبيرين منقولين، فإن T1+T2 وT1*3 ليست لا مطلقة ولا منقولة .

ولنتأكد من ذلك يكفي أن نقوم بإجراء عملية نقل بمقدار 100 مثلاً :

T1	تصبح	T1 + 100
T2	تصبح	T2 + 100
T1 + T2	تصبح	T1 + T2 + 100
T1 * 3	تصبح	T1 * 3 + 300

التعابير لا تحتمل نفس الإزاحة .

إستعمال التعابير هو بشكل خاص مفيد لأنه يسمح بتحديد العناصر حيث القيم هي قابلة للتغيير عند التأويل وذلك بشكل مُعامِلات ومتغيرات (مثلاً صفحة 122، السطر 78 من البرنامج) . كل تعديل في قيمة المتغيّر من التعبير سيكون محسوباً من جديد بواسطة المؤوّل وليس بواسطة المبرمج ، مما يُسهِّل عمل المبرمج .

8 توجيهات تعريف الرموز

لنأخذ هذه القطعة من برنامج بلغة فورتران :

```
      DIMENSION TAB(100)
      ---------
      DO 50 I=1,100
50    TAB(I)=I
      ---------
```

يطلب الأمر DIMENSION حجز 100 كلمة ـ ذاكرة بمجموعة تحت إسم الجدول TAB . تدل القواعد الضمنية المتعلقة بنوع المعرِّفات أنّ هذا الجدول سيتألف من أعداد حقيقية ، أي مكوَّدة في التمثيل بفاصلة متحركة بدقة بسيطة . يعرف المصرِّف بأنه يجب أن يستعمل ، لتوليد كود التعليمات الحسابية التي تبلغ TAB ، التعليمات الحسابية بدقة بسيطة .

وفي فورتران ، كما في جميع لغات البرمجة ، كل رجوع إلى معرِّف يفترض أن يكون الأخير معروفاً من المصرِّف ، أي مُحدَّداً خلال البرنامج بواسطة نوعه (حقيقي ، صحيح . .) وطوله مُقاساً بالكلمات أو بالبايتات . وفي النهاية يخصِّص المعرِّف TAB بخاصية ـ قيمة (قيمة المعرِّف ستكون عنوانه) ،وبخاصية ـ طول (بعد الحيِّز المشار إليه بالبايتة) .

في لغة التأويل المسألة هي نفسها ، يجب أن يحدَّد كل رمز بواسطة خواصّه . سنرى توجيهين DC وDS يسمحان بتعريف الثوابت وحجز مكان من الذاكرة ، والتوجيه EQU الذي يسمح بإجراء توازنات بين الرموز .

1.8 . تعريف الثابتة DC

كثير الإستعمال ، هذا التوجيه يسمح بحجز منطقة من الذاكرة تحتوي على القيمة المدعوّة ثابتة وبتسميتها بواسطة أحد الرموز .

شكل هذا التوجيه هو التالي :

عامل	كود العملية	رمز
d t m 'c'	DC	[وسم]

- الوسم هو الإسم الرمزي للثابتة وهو إختياري .
- d هو عامل الازدواجية ، وهو اختياري ، وإذا كان مهملاً فإن قيمته تعادل 1 . إنّه يشير إلى العدد الذي يجب أن تولّد فيه الثابتة .
- t هو النوع ، يمكن أن يكون أحد الأكواد الموجودة في الجدول التالي :

الاصطفاف	الطول الضمني	نسق المكنة	نوع الثابتة	كود
باية		EBCDIC	سمة	C
باية		ثنائي بفاصلة ثابتة	سادس عشري	X
باية		ثنائي	ثنائي	B
كلمة	كلمة واحدة	كلمة ثنائية بفاصلة ثابتة	عشري	F
نصف كلمة	نصف كلمة	نصف كلمة بفاصلة ثابتة	عشري	H
كلمة	كلمة واحدة	فاصلة متحرّكة ودقّة بسيطة	عشري	E
كلمة مزدوجة	كلمتان	فاصلة متحرّكة ودقّة مضاعفة	عشري	D
كلمة مزدوجة	4 كلمات	فاصلة متحركة ودقّة رباعية	عشري	L
باية		عشري موسع	عشري	Z
باية		عشري مكثّف	عشري	P

جدول 1.8

في المكنة تُحصر الثوابت في حدود البايتة ، نصف الكلمة ، الكلمة أو الكلمة المزدوجة حسب نوعها ما عدا في الحالة التي نُحدِّد فيها طولها (أو نستعمل معدِّلاً للطول) .

- m هو معدِّل طول الثابتة ، ويمكن أن يكون :

أ ـ معدل طول ضمني يُكتب على شكل Ln حيث n هو عدد البايتات في التمثيل الداخلي . إنّ وجود معدِّل للطول يُصفِّر قاعدة الاصطفاف الضمنية .

ب ـ مُعدِّل للحصر يُكتب على الشكل التالي : Sn .

معدِّل الحصر يقوم بإجراء إزاحة لِـ n بتة إلى اليسار إذا كانت n إيجابية ، وإلى اليمين إذا كانت n سلبية . أي يقوم بإجراء ضرب أو قسمة صحيحة على 2^n .
معدِّل الحصر ، ويدعى أيضاً المقياس ، يُطبَّق على الثوابت E ، D وL .

'c' هي الثابتة المحدَّدة بين فاصلتين عليين (' ') . الثوابت يُمكن أن تكون محدَّدة بإشارة ، فاصلة عشرية وبأسّ (قوة) يُرمز إليه بالحرف E . الأمثلة التالية تُظهر لنا مختلف الإمكانيات . وهناك جدول في الملحق يُوجز لنا مميزات الثوابت .

```
  LOC  OBJECT CODE    ADDR1 ADDR2  STMT    SOURCE STATEMENT
000000                                1 CONST    CSECT
                                      2          PRINT DATA
                                      3 *---------------------------------------------------------------------------
                                      4 *
                                      5 * CONSTANTES DE CARACTERES.PAS D'ALIGNÉMENT PARTICULIER. LONGUEUR 256
                                      6 * CADRAGE A GAUCHE. TRONCATURE A DROITE
000000 C1C2C3C4C5                     7 CARACT   DC    C'ABCDE'
000005 E3D9D6                         8          DC    CL3'TRONCATURE A DROITE'
000008 C1C2C3404040                   9          DC    CL6'ABC'  CADRAGE A GAUCHE COMPLETE PAR DES BLANCS
00000E D37DC1D7D6E2E3D9              10          DC    CL8'L''APOSTROPHE'       UN SEUL APOSTROPHE ENGENDRE
000016 C150C2                        11          DC    CL3'A&&B'                MEME REMARQUE
000019 C1C2C3C4C1C2C3C4              12          DC    3CL4'ABCDE'              REPETITION ET TRONCATURE
000021 C1C2C3C4
                                     13 *---------------------------------------------------------------------------
                                     14 *
                                     15 * CONSTANTES HEXADECIMALES. CADRAGE A DROITE.  TRONCATURE A GAUCHE.
000025 0F01A2                        16 HEXA     DC    X'F01A2'                 LONGUEUR IMPLICITE
000028 000001AB0C                    17          DC    XL5'1AB0C'               LONGUEUR EXPLICITE
00002D 1FBC1FBC1FBC                  18          DC    3XL2'A1FBC'              TRONCATURE
                                     19 *---------------------------------------------------------------------------
                                     20 *
                                     21 * CONSTANTES BINAIRES. LONGUEUR MAXI 256 OCTETS , CADRAGE A DROITE
                                     22 * COMPLETE PAR DES ZEROS A GAUCHE. ALIGNEMENT SUR L'OCTET.
000033 11DA                          23 BINAIRE  DC    B'1000111011010'
                                     24 * TRONCATURE A GAUCHE.
000035 39                            25          DC    BL1'1000111001'          TRONCATURE
000036 000600060006                  26          DC    3BL2'110'                REPETITION
                                     27 *---------------------------------------------------------------------------
                                     28 *
                                     29 * CONSTANTES EN VIRGULE FIXE SUR UN MOT (F) OU UN DEMI-MOT (H).
                                     30 * ALIGNEMENT SUR LE MOT OU LE DEMI-MOT. LORSQU'UNE LONGUEUR EST
                                     31 * SPECIFIEE IL N'Y A PLUS D'ALIGNEMENT. LA CONSTANTE EST EN DECIMAL
00003C 138803E80011                  32 HALFWORD DC    H'5E3,10E2,17.2'
000042 FFFF                          33          DC    H'-1'
000044 0190                          34          DC    HS3'50'                  DECALAGE 3 BITS A GAUCHE (*8)
000046 0190                          35          DC    H'400'
000048 0006                          36          DC    HS-3'50'                 DECALAGE 3 BITS A DROITE (/8)
00004A 0006                          37          DC    H'6'
00004C 0000007D                      38 FULLWORD DC    F'125'
000050 0000000D                      39          DC    F'12.5'                  ARRONDI SUPERIEUR
000054 0000000C                      40          DC    F'12.3'                  ARRONDI INFERIEUR
000058 000004E2                      41          DC    F'12.5E2'
00005C 0000190019                    42          DC    FL3'25',FL2'25'          MODIFICATION LONG ET ALIGN.
000061 000000
000064 00000064                      43          DC    FS2'25'                  DECALAGE A GAUCHE DE 2 BITS
000068 00000000                      44          DC    F'0'
                                     45 *---------------------------------------------------------------------------
```

```
  LOC  OBJECT CODE      ADDR1 ADDR2  STMT   SOURCE STATEMENT
                                       47 *-----------------------------------------------------------------
                                       48 *
                                       49 * CONSTANTES FLOTTANTES EN SIMPLE PRECISION. ALIGNEMENT SUR LE MOT
                                       50 * CADRAGE A DROITE. PAS DE TRONCATURE, VALEUR ARRONDIE.
                                       51 * LONGUEUR IMPLICITE 4 OCTETS.
00006C 42190000                        52 FLOTTANT DC      E'25'
000070 C2190000                        53          DC      E'-25'
000074 00000000                        54          DC      E'0'
000078 C1100000                        55          DC      E'-1'
                                       56 *-----------------------------------------------------------------
                                       57 * CONSTANTES FLOTTANTES EN DOUBLE PRECISION
                                       58 * ALIGNEMENT SUR LE DOUBLE-MOT. CADRAGE A DROITE, PAS DE TRONCATURE
                                       59 * VALEUR ARRONDIE. LONGUEUR IMPLICITE 8 OCTETS
00007C 00000000
000080 4219000000000000                60 DOUBLE   DC      D'25'
000088 C219000000000000                61          DC      D'-25'
000090 32170EF54646D497                62          DC      D'125E-20'
000098 B2170EF54646D497                63          DC      D'-125E-20'
                                       64 *-----------------------------------------------------------------
                                       65 *
                                       66 * CONSTANTES FLOTTANTES EN QUADRUPLE PRECISION.
                                       67 * ALIGNEMENT SUR LE DOUBLE-MOT. LONGUEUR IMPLICITE 16 OCTETS.
                                       68 * PAS DE TRONCATURE. LA VALEUR EST ARRONDIE. EXPOSANT DE -85 A +75.
0000A0 C27D000000000000                69 QUADRU   DC      L'-125'
0000A8 B400000000000000
0000B0 4320D00000000000                70          DC      L'525'
0000B8 3500000000000000
                                       71 *-----------------------------------------------------------------
                                       72 *
                                       73 * CONSTANTES DECIMALES DITES "ZONED". LONGUEUR MAXI
                                       74 * DE 16 OCTETS. LE SIGNE EST SITUE DANS LE QUARTET DE GAUCHE
                                       75 * DU DERNIER OCTET A DROITE.  CADRAGE A DROITE, TRONCATURE A GAUCHE.
                                       76 * X'F' OU X'C' DANS LA POSITION DE SIGNE SONT CONSIDERES COMME +
                                       77 * X'D' OU X'E' DANS LA POSITION DE SIGNE SONT CONSIDERES COMME -
                                       78 * LE POINT DECIMAL N'EST JAMAIS TRADUIT EN BINAIRE.
                                       79 * CADRAGE A DROITE, TRONCATURE A GAUCHE.
0000C0 F1F2F3F4C5                      80 ZONED    DC      Z'12345'             LONGUEUR IMPLICITE
0000C5 F1F2F5C2                        81          DC      Z'+125.2'
0000C9 F2F3F4F5C6                      82          DC      ZL5'+123456'         TRONCATURE A GAUCHE
0000CE F0F0F1D2                        83          DC      ZL4'-12'
                                       84 *-----------------------------------------------------------------
                                       85 *
                                       86 * CONSTANTES DECIMALES CONDENSEES DITES "PACKED".
                                       87 * MEMES REGLES QUE POUR LES CONSTANTES ZONEES.
                                       88 * LE SIGNE EST SITUE DANS LE DERNIER QUARTET DE DROITE.
0000D2 01234C                          89 PACKED   DC      P'1234'
0000D5 00000001234D                    90          DC      PL6'-1234'
0000DB 025D025D025D                    91          DC      3P'-25'
                                       92 *-----------------------------------------------------------------
                                       93          END
```

2.8 . ثوابت العنوان[1]

إنّ تعريف ثابتة ـ عنوان يعني حجز مكان من الذاكرة لتخزين عنوان أحد العناصر . نشير هنا إلى بعض المفاهيم الأساسية . العنوان الفعلي ، أي العنوان الحقيقي لأحد العناصر هو غير معروف إلا عند شحن البرنامج في الذاكرة . لذا فمن غير الممكن ، في مرحلة التأويل والتجميع ، أن يكون بتصرفنا العنوان الفعليّ الخاص بالرمز . نبلغ الرمز بواسطة الإزاحة نسبة إلى مضمون مرصف القاعدة .

في بعض الأحيان يبدو من غير الممكن بلوغ أحد الرموز التي لا تنتمي إلى الزجلة التي تكون في طور المعالجة من قِبل المؤول . هذه هي الحالة ، مثلاً ، عندما نرغب بإجراء تفريع إلى برنامج ـ ثانوي مؤول ومترجم على حدة . الحلّ يقوم إذاً ، بالنسبة للمؤول ، على بلوغ مباشر بسبب وجود كلمة ، تدعى ثابتة ـ عنوان ، يقوم الشاحن (Loader) بملئها بشكل مناسب .

مثلاً :

نرغب ، للتفريع إلى المرصف 15 ، شحنه بعنوان نقطة الدخول P1 لبرنامج ـ ثانوي مؤول على حدة . سنحفظ ، في الزجلة المُنادية ، كلمة تدعى هنا ADRP1 سيتم تعريفها كثابتة عنوان خارجية . والمؤول سيقوم بإعدادها وتصفيرها ، كما سيقوم الشاحن بتخزينّ العنوان الفعلي P1 في داخلها . العنوان P1 سنحصل عليه إذاً في المرصف 15 بواسطة التعليمة :

. L 15, ADRP1

إنّ نسق تعريف ثابتة العنوان هو التالي :

عامل	كود ـ العملية	رمز
d t m ׀(c)	DC	[وسم]

نسق هذا الأمر لا يتميَّز عن نسق تعريف الثوابت إلا بتبديل الفواصل العليا بالأهلّـة .

- d هو عامِل الإزدواجية ، وإذا جرى إهماله فإنّـه يعادل 1 .
- t هو كود نوع الثابتة .

(1) دراسة هذه الفقرة المفيدة للفهم الكامل يمكن أن يقفز عنها عند القراءة الأولى .

وقد يكون A ، Y ، S ، V أو Q (النوع Q ليس متوفـراً سوى تحت النظام OS) . النوعان A وY يسمحان بتعريف الثوابت بواسطة تعابير بسيطة أو مركّبة ، مطلقة أو منقولة . القيمة ثابتة العنوان محدَّدة لجهة اليمين في كلمة (نوع A) أو نصف كلمة (نوع Y) . الثوابت من نوع S تسمح بتخزين عناوين بشكل قاعدة وإزاحة على نصف كلمة . ولا يمكنها أن تعرَّف في نص حرفي . تستعمل الثوابت من نوع V لتعريف عناوين خارجية من نوع « إسم برنامج ثانوي » .

- m هو عبارة عن معدِّل الطول الضمني . وجود المعدِّل يؤدي إلى إلغاء قاعدة الاصطفاف الأوتوماتيكية (alignement) .
- C هو عبارة عن الثابتة نفسها مكتوبة بداُخل أهلَّـة . الأمثلة في الصفحة 75تعرض وتعرِّف كل نوع من الثوابت .

استعمال ثابتة العنوان :

تُستعمل :

- لشحن عنوان في مرصف .
- لاجراء وصلات بين البرنامج والبرنامج الثانوي .

وسيتم درسُ ذلك في الفصلين 20 و21 .

3.8 . أمر حجز مواقع من الذاكرة

هذا الأمر هو عبارة عن توجيه يسمح بحجز موقع من الذاكرة دون إعداد أو تهيئة مضمونه عند التأويل . هذا الأمر يُؤدي إذاً إلى زيادة مضمون عداد المواقع . ويسمح بتسمية المناطق المحدَّدة وبلوغها رمزياً . النحو ، القريب من نحو التوجيه DC ، هو التالي :

رمز	كود العملية	عامل
[وسم]	DS	d t m

- d مُعامِل الازدواجية ، وهو اختياري . وإذا كان صفراً فهو يسمح بزيادة عدّاد المواقع حتى حدود نصف كلمة ، كلمة أو كلمة مزدوجة حسب نوع t المرتبطة بالمنطقة . هذه الخصوصية تستعمل كثيراً ونوضّـحها في الأمثلة والأسئلة . سنشير هنا ، إلى أنه مع وجود عامِل إزدواجية يعادل صفراً ، فإن الوسم الموجود في منطقة الرمز هو مخزَّن في جدول الرموز .

```
  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

000000                                1            CSECT
                                      2            EXTRN  SYMBEXT                  SYMBOLE EXTERNE
000000                        00000   3            USING  *,12
                              003E8   4            ORG    *+1000
                              000FF   5  ABSOL     EQU    255
0003E8 FFFFFFFF                       6  RELOC     DC     X'FFFFFFFF'              SYMBOLE TRANSLATABLE
                                      7  *--------------------------------------------------------------------
                                      8  *
                                      9  * CONSTANTES D'ADRESSE DE TYPE A.
                                     10  * S'ECRIT DC  A(EXPRES.ABSOLUE OU TRANSLATABLE)
                                     11  * ALIGNEMENT SUR LE MOT. LONGUEUR IMPLICITE 4 OCTETS.
                                     12  * LONGUEURS EXPLICITES POSSIBLES DE 1 A 4 OCTETS.
                                     13  * TRONCATURE A GAUCHE. PEUT ETRE DEFINIE DANS UN LITERAL.
                                     14  *
0003EC 000003E8                      15  A         DC     A(RELOC)
0003F0 00000113                      16            DC     A(ABSOL+20)
0003F4 000003F4                      17            DC     A(*)
0003F8 7D16                          18            DC     AL1(125,22)              LONGUEUR EXPLICITE
0003FA 0000
0003FC 000003F2                      19            DC     A(RELOC+10)
000400 00000000                      20            DC     A(SYMBEXT)               SYMBOLE EXTERNE
                                     21  *--------------------------------------------------------------------
                                     22  *
                                     23  * CONSTANTES D'ADRESSE DE TYPE Y
                                     24  * S'ECRIT DC  Y(EXPRES. ABSO CU TRANSLATABLE)
                                     25  * ALIGNEMENT SUR LE DEMI-MOT. LONGUEUR IMPLICITE 1/2 MOT.
                                     26  * LONGUEURS EXPLICITES POSSIBLES DE 1 OU 2 OCTETS.
                                     27  * TRONCATURE A GAUCHE. PEUT ETRE DEFINIE DANS UN LITERAL.
                                     28  *
000404 00000002                      29  B         DC     Y(*-B,*-B)               NOTER QUE LES 2 * SONT
                                     30  *                                         EGAUX A B ET B+2
000408 7D                            31            DC     YL1(125)                 LONGUEUR EXPLICITE
000409 02                            32            DC     YL1(258)                 TRONCATURE A GAUCHE
                                     33  *--------------------------------------------------------------------
                                     34  *
                                     35  * CONSTANTES D'ADRESSE DE TYPE S.
                                     36  * S'ECRIT        DC   S(EXPRESSICN ABSOLUE)
                                     37  *       OU       DC   S(EXPRESSICN TRANSLATABLE)
                                     38  *       OU       DC   S(EXPR. ABS(EXPR. ABS))
                                     39  * EST ASSEMBLEE DANS UN 1/2 MCT. ALIGNEE SUR LE 1/2 MOT.
                                     40  * NE PEUT ETRE DEFINIE DANS UN LITERAL.
                                     41  *
00040A 0400                          42  C         DC     S(1024)                  BASE 0, DEPL =1024
00040C C3E8                          43            DC     S(RELOC)                 BASE ET DEPLAC DE RELOC
00040E C200                          44            DC     S(512(12))               (DEPLAC(BASE))
                                     45  *--------------------------------------------------------------------

                                     47  *--------------------------------------------------------------------
                                     48  *
                                     49  * CONSTANTES D'ADRESSE DE TYPE V.
                                     50  * UTILISEES SEULEMENT POUR LES ADRESSES EXTERNES DE TYPE NOM-DE-PROG.
                                     51  * S'ECRIT DC  V(1 SYMBOLE TRANSLATABLE EXTERNE)
                                     52  * LE SYMBOLE TRANSLATABLE NE FIGURE PAS DANS UN ORDRE EXTRN.
                                     53  * LONGUEUR IMPLICITE 4 OCTETS, MODIFICATEUR DE LONGUEUR = 3 OU 4.
                                     54  * ALIGNEMENT SUR UNE FRONTIERE DE MOT  PEUT APPARAITRE DANS UN LITERAL
                                     55  * L'ASSEMBLEUR GENERE UN MOT NUL.
                                     56  *
000410 00000000                      57  D         DC     V(ENTREESP)
000414 00000000                      58            DC     V(SOUSPROG)
                                     59  *--------------------------------------------------------------------
                                     60            END
```

ـ t يُحدِّد نوع المنطقة أي بالتحديد كما جرى بالنسبة للأمر DC . وهو إلزامي ويحدِّد التسطير الضمني .

ـ m هو معدَّل الطول ويُكتب Ln ، حيث n هو طول المنطقة بالبايتات . كما بالنسبة للأمر DC فهو إختياري .وجوده يلغي فعل الإصطفاف الضمني . سنشير هنا إلى أن الطول الأقصى للثابتة من نوع سلسلة السمات المُحدَّدة في الأمر DC هو 256 بايتة ، وإستعمال النظام OS يسمح بـ 65535 بايتة .

لتسهيل صيانة البرامج سنستعمل : ETIQ DS 0H لتعريف نقاط التفريع .

قدر المستطاع سنفضل إستعمال الأمر DC عن الأمر DS الذي يقوم بإعداد المنطقة بقيمة محايدة ستكون مرئية في عملية DUMP (دلق) .

4.8 . توجيه التعادل EQU

يسمح بتعريف رمز وإعطائه قيمة مطلقة أو محوَّلة ويُكتب على الشكل التالي :

رمز (Symbol)	EQU	تعبير مطلق أو محوَّل

سنشير هنا إلى أن وجود الرمز هو إلزامي . لا يحجز التوجيه أي موقع من الذاكرة ولا يقوم سوى بإنشاء رمز جديد في جدول الرموز . ويمكن أن يكون موجوداً في أي موقع من البرنامج ويُستخدم :

1 ـ لاستعمال أسماء بدلاً من القيم . تجري العادة مثلاً على كتابة :

```
R0      EQU  0
R1      EQU  1
--      ---  -
R15     EQU  15
```

مما يسمح ، منذ البداية ، ببلوغ المراصف بواسطة الأسماء R0 ، R1 ، . . . ، R15 . بدلاً من القيم 0 ، 1 ، . . . ، 15 . هذا ما يؤدي إلى فائدة ووضوح في العمل ولكن أيضاً إلى إمكانية إيجاد مراجع المراصف بسهولة لأنها ستظهر في جدول الرموز وفي البلوغ التصالبي .

2 ـ لتخصيص قيمة جديدة محدَّدة داخل البرنامج لرمز معين ، أي معرَّف خلال الأسطر السابقة .

```
RO       EQU  0
REGO     EQU  RO        ( رمز مطلق )
----     ---  --
DEB      LR   R1,R2
----     ---  --
DEBUT    EQU  DEB       ( رمز محوَّل )
----     ---  --
ZONE     DS   4F
Z1       EQU  ZONE+12   ( تعيير محوَّل )
```

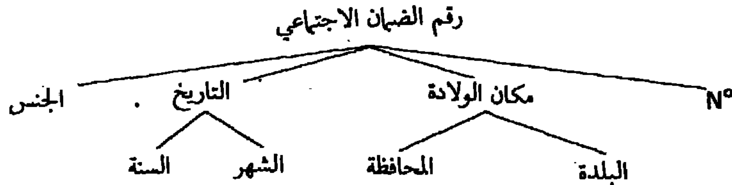
3 ـ لحساب التعابير حيث القيمة مجهولة في لحظة الكتابة أو صعبة الحساب وتخصيص رمز لها .

```
EXPRES   EQU  A-(B+C)/5-D
ETIQ     EQU  *              ( قيمة عداد المواقع )
```

تمارين

تمرين 1.8 ـ ولّد ، بواسطة تعريف ثابتة مخصصة ، منطقة من الذاكرة بحجم 100 بايتة تحتوي على سلسلة من 100 عدد صحيح طبيعي . نفس السؤال لمنطقة بحجم 100 كلمة .

تمرين 2.8 ـ عرِّف حجز من الذاكرة لاستيعاب رقم الضمان الإجتماعي (13 سمة) مع وصف للهيكلية التالية .



وذلك بفحص الخاصية ـ طول لكل معرَّف مذكور .

تمرين 3.8 ـ باستعمال الأمر ORG (فقرة 3.20) ، مطلوب تعريف منطقة من الذاكرة يمكن أن تستوعب إما ثمناً (8 أرقام عشرية موسّعة) أو كمية (4 أرقام عشرية موسعة) ، أو رقماً (عدداً صحيحاً بفاصلة ثابتة) ونصاً من 10 سمات . يتعلّق ذلك بإعادة تعريف من نوع REDEFINES بلغة كوبول .

9 كتابة العناوين بلغة المؤول

1.9 . قاعدة ضمنية ، قاعدة جليّة

في جسم التعليمات الآلية ، فإن العناوين المحوّلة تكون ممثّلة بواسطة مرصف قاعدي ، وإزاحة ومرصف دليل (حالة النسق RX) . عند كتابة التعليمات - الآلية بلغة المؤول سنقوم بإيجاد ثلاثة متأثرات . لقد لاحظنا حتى الآن أنه كان يوجد ستة أنسقة مختلفة للتعليمات الآلية . إضافة لذلك ، وفي لغة المؤول ، فإن كتابة منطقة العوامل (منطقة العناوين والثوابت) ستتغيّر حسب نسق المكنة .

لنأخذ تعليمة شحن المرصف 3 (LOAD) من خلال مضمون عنوان معين . لنفترض إن المرصف 15 قد جرى إختياره كمرصف قاعدي ، وإن العنوان موضع السؤال هو موجود على مسافة 512 (في القاعدة العاشرة) من العنوان القاعدي وهو مؤشر بواسطة المرصف 5 . التعليمة - الآلية سيكون لها الشكل التالي :

5	8	3	5	F	2	0	0
COP		R_1	X_2	B_2	D_2		

سيكون بإمكان المبرمج بلغة المؤول أن يكتب التعليمة على الشكل التالي : L 3 , 512 (5, 15) . القاعدة 15 هي هنا مسماة بشكل واضح . لا نرى بهذا الشكل الفائدة الرمزية من لغة المؤوّل .

لتأمين بساطة أكبر فإن المؤول يسمح بعدم ذكر القاعدة في منطقة العوامل التابعة للتعليمة . يكفي لذلك أن نصرّح ، بواسطة التوجيه USING * ,15 ، أن التعليمات التالية يجب أن تؤول (تجمّع) مع المرصف 15 كقاعدة . الفائدة الأولى هي السماح بتعديل مرصف القاعدة دون إعادة كتابة جميع التعليمات . كذلك ، فإن الإزاحة ومرصف المؤشر يمكن أن يتم تمثيلهما بشكل رمزي عند الحاجة . هكذا ، فلنأخذ العنوان المحوّل ALPHA الموجود على المسافة 512 بايتة من العنوان القاعدي . ولنشحن في

المرصف 3 مضمون العنوان ALPHA المؤشر بواسطة المرصف 5 . بإمكاننا كتابة التعليمات التالية بلغة المؤول :

- بتحديد القاعدة بشكل واضح : (5,15) L 3,512 .
- أو (5) L 3, ALPHA ، القاعدة هي ضمنياً مرتبطة بـ ALPHA ومحدّدة بواسطه المؤول حسب التوجيه USING . يوجد عدة إمكانيات لكتابة منطقة العوامل ، وهذا ما سنقوم بشرحه الآن .

2.9 . كتابة العوامل

في الإعتبارات التالية D ، X ، B ، R ، M وL تُمثّل على التوالي الإزاحة ، رقم مرضف المؤشر ، رقم مرصف القاعدة ، رقم المرصف العام ، قناع (موجود في التعليمة) والطول . الدلائل 1 ، 2 و3 هي مرتبطة بمختلف المتأثرات . جميع هذه الرموز يجب أن تكون عبارة عن تعابير مطلقة . S ستمثّل تعبيراً متحوّلاً يمكن أن يُختزل عملياً إلى رمز واحد . وبتحديد أكثر للمرصف القاعدي ، فإنّ عوامل (متأثرات) التعليمات يُمكن أن تُكتب بلغة المؤول ، حسب النسق ، على الشكل التالي :

النسق	المُعاملات
RR	R_1,R_2
RX	$R_1,D_2(X_2,B_2)$
RS	$R_1,R_3,D_2(B_2)$ $R_1,M_3,D_2(B_2)$
SI	$D_1(B_1),I_2$
SS	$D_1(L,B_1),D_2(B_2)$ $D_1(L_1,B_1),D_2(L_2,B_2)$
S	$D_2(B_2)$

جدول 1.9

العنوان المحوّل هو دائماً العنوان المحسوب في لحظة تنفيذ الجمع D+X+B للتعليمات ذات النسق RX أو D+B للتعليمات RS ، SI أو SS . العوامل D(X,B) ، D(B) ، وD(L,B) يُمكن أن تُستبدل بواسطة العوامل حيث رقم المرصف القاعدي والإزاحة سيتم حسابها بواسطة المؤول . وستُكتب إذاً على الشكل التالي : S ، S(X) ، أو S(L) .

الجدول التالي يعرض لنا مختلف إمكانيات كتابة هذه المُعاملات حسب نسق

التعليمة . سنشير هنا إلى أنه بداخل الأهلة ، وفي الشكلين مع قاعدة ضمنية أو جلية ، لا يمكن أن نجد سوى التعابير المطلقة حيث المعنى الأساسي ، الدليل أو الطول يتعلّق بنسق التعليمة وبالطبيعة مطلق أو محوّل للتعبير المذكور على يسار الأهلّة .

أمثلة :

ABS و TRANS هما تعبيران مطلقان ومحوّلان . (ABS2) ABS1 في التعليمة RS يُمكن أن تُفهم وكأنها (D(B . (TRANS) ABS1 هي مغلوطة مهما يكن النسق ، (TRANS (ABS1 يجب أن تُفهم كأنها (S(X في التعليمة RX وكأنها (S(L في التعليمة SS .

نسق التعليمة	الكتابة بتعابير مطلقة قاعدة جلية	الكتابة بتعابير محوّلة قاعدة ضمنية
RS et SI	D(B)	S
SS	D(L,B) D(,B) (1) D(B)	S(L)
RX	D(X,B)	S(X) S

جدول 2.9

حالات خاصة

X أو B يعادل صفراً .

D(0) يُمكن أن يُكتب D

D (0,B) يمكن أن يُكتب D(,B)

D (X,0) يمكن أن يُكتب D(X,) أو D(X) . (أمثلة أنظر صفحة 82) .

3.9 . قواعد الاصطفاف أو التراصف

مع أن أوالية العنونة تسمح بعنونة البايتة ، فإن عناوين متأثرات التعليمة يجب أن تخضع لبعض قواعد التوافق . قواعد كهذه هي موجودة على جميع المكنات .

تستعمل التعليمات متأثرين قد يكونان عبارة عن مرصف وعنوان من الذاكرة أو عنوانين من الذاكرة . تحدّد القواعد حسب المعطيات التي تُعالجها التعليمات .

بالنسبة للتعليمات التي تُعالج كلمات ـ مزدوجة ، كلمات أو نصف ـ كلمات ، فإن

(1) الطول هو ضمني ، المؤوّل يختار الخاصّية ـ طول . الطول المؤوّل هو دوماً الطول الفعلي ناقص واحد .

```
  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                     1             PRINT DATA
000000                               2             START 0
                                     3  * SEQUENCE D'ENTREE
000000 90EC D00C      0000C          4             STM    14,12,12(13)      SAUVEGARDE DES REGISTRES DU
000004 05C0                          5             BALR   12,0              CHARGEMENT DU REGISTRE DE BASE
                            00006    6             USING  *,12              R12 = REGISTRE DE BASE
                                     7  *                                   PROGRAMME APPELANT
000006 47F0 C00E      00014          8  ACBASE     BC     15,INSTR1
                                     9  *
                            00000   10  R0         EQU    0
                            00003   11  R3         EQU    3
                            00005   12  R5         EQU    5
                            0000C   13  R12        EQU    12
00000A 0700                         14             CNOP   0,4               ALIGNEMENT MOT
00000C 01234567                     15  ALPHA      DC     X'01234567'
000010 89ABCDEF                     16             DC     X'89ABCDEF'
                                    17  *
000014 5835 C004      00004         18  INSTR1     L      3,4(5,12)         (1) BASE 12 EXPLICITE
000018 5835 C004      00004         19             L      R3,4(R5,R12)      (2) BASE 12 EXPLICITE
                                    20  *
                                    21  * TOUTES LES INSTRUCTIONS (3) A (7) CHARGENT X'89ABCDEF' DANS LE
                                    22  * REGISTRE 3. L'ECRITURE (3) EST LA SEULE QUI SOIT INDEPENDANTE
                                    23  * DE LA PLACE DE ALPHA PAR RAPPORT A L'ACRESSE DE BASE.
00001C 5830 C00A      00010         24             L      R3,ALPHA+4        (3) EMPLOI D'UNE EXPRESSION
                                    25  *                                   TRANSLATABLE. BASE IMPLICITE.
000020 5830 C008      00008         26             L      3,8(0,12)         (4) INSTR. IDENTIQUE SYMB. ABSOL
000024 5830 C008      00008         27             L      3,8(,12)          (5)
000028 5830 C008      00008         28             L      R3,8(R0,R12)      (6)
                                    29  *
00002C 5830 C052      00058         30             L      R3,=X'89ABCDEF'   (7) EMPLOI D'UN LITTERAL
                                    31  *
000030 5830 0008      00008         32             L      R3,8              (8)  "8" EST UN DEPLACEMENT
                                    33  *
000034 5830 C007      0000D         34             L      R3,ALPHA+1        (9) ERREUR D'ALIGNEMENT
          *** ERROR ***
                                    35  *
000038 5830 000C      0000C         36             L      R3,(12)           (10) "12" EST UN DEPLACEMENT
00003C 5830 C000      00000         37             L      R3,0(,12)         (11) "12" EST UN REG DE BASE
000040 0000 0000      00000         38             L      R3,0(12,)         (12) ERREUR DE SYNTAXE
          *** ERROR ***
000044 0000 0000      00000         39             L      R3,(12,)          (13) ERREUR DE SYNTAXE
          *** ERROR ***
000048 0000 0000      00000         40             L      R3,(,12)          (14) ERREUR DE SYNTAXE
          *** ERROR ***
00004C 583C 0000      00000         41             L      3,0(12)           (15) 12 EST UN INDEX

                                                ASSEMBLER DIAGNOSTICS AND STATISTICS
                             STMT  ERROR CODE   MESSAGE

                              34   IFC220       ALIGNMENT ERROR NEAR OPERAND COLUMN 11
                              38   IFO234       PREMATURE END OF EXPRESSION NEAR OPERAND COLUMN 9
                              39   IFO234       PREMATURE END OF EXPRESSION NEAR OPERAND COLUMN 7
                              40   IFO234       PREMATURE END OF EXPRESSION NEAR OPERAND COLUMN 5

                             NUMBER OF STATEMENTS FLAGGED IN THIS ASSEMBLY =      4
                             HIGHEST SEVERITY WAS    8
```

عناوين المتأثرات يجب أن تُصفّ حسب الحدود المناسبة . أما تلك التي تعالج السمات فلا يوجد أية مشكلة بالنسبة لها . إنّ عدم المحافظة على هذه القواعد يؤدّي الى حدوث مشكلة في المؤول (أنظر المثل السطر 33) ، فهو يؤدي عند التنفيذ إلى انقطاع من نوع «Specification» (تميز) . التعليمات يجب أيضاً أن تُصفّ في حدود نصف كلمات .

تمارين

تمرين 1.9 ـ للتعليمات أدناه :

1 ـ إفحص إذا كانت العناصر التي تؤلف المتأثرات هي مُطلقة أو محوَّلة .

2 ـ باعتماد النسق المرتبطة بكل تعليمة نستخلص ، فقط حسب المعايير النحوية ، إذا كانت التعليمات صحيحة .

3 ـ قم بإجراء تأويل التعليمات الصحيحة .

```
        CSECT
        USING   *,12           مرصف القاعدة = 12 ,
ADBASE  L       B,D            RX النسق
        L       3,D(3)         RX
        LR      A,D            RR
        ST      D,X'4'(3,C)    RX
        L       A,B'1011'(3)   RX
        L       D,E(B)         RX
        L       A,E(B)         RX
        MVC     A(B,C),D       SS
        MVC     E(L'D),D       SS
        L       2,D+L'D        RX
A       EQU     0
B       EQU     1
C       EQU     10
D       DS      5F
E       DS      12F
        END
```

10 . التعليمات بلغة المؤول
عموميات

سنقوم بدراسة التعليمات ـ الآلية حسب نوع التمثيل الداخلي الذي تُعالجه هذه التعليمات . من البديهي أن تكون التعليمات الحسابية العشرية ، مثلاً ، بدون معنى إلا عندما نُقدِّم لها معطيات مكوَّدة عشرياً . مثلاً ، من الواضح أن المراصف المبلوغة بالتعليمات المتحركة هي مراصف متحركة .

سنبدأ بالتعليمات التي تعمل على المراصف العامة ، ولكن في البداية يجب عرض الترميز المعتمد .

1.10 . الترميز

سيتم تحديد التعليمات ـ الآلية حسب النسق التالي :

المعنى	كود العمليات سادس عشري	النسق	العوامل	كود ـ العملية الرمزي
LOAD $(S_2) \to R_1$	COP=58	RX	$R_1,D_2(X_2,B_2)$	L

تشير العوامل إلى العناوين مع قاعدة محدَّدة بشكل جلي . أمّا الشروحات فتذكر هذا العنوان بشكل رمزي . فإذاً S_2 ستعني العنوان المحسوب بإضافة مضمون المراصف القاعدية والمؤشر إلى الإزاحة . في المجموع فإن $S_2=D_2+B_2+X_2$ بالنسبة للتعليمات RX و$S_2=D_2+B_2$ للباقية .

سنجد في الحيِّز مُعاملات أو في الشروحات الرموز التالية :

R_1, R_2 هي عبارة عن أرقام المراصف التي يمكن أن تُستبدل بالتعابير المطلقة .
D قيمة الإزاحة بالنسبة إلى العنوان القاعدي .
X رقم المرصف المؤشر المُستعمل .
B رقم المرصف القاعدي .
M قناع من أربع بتات موجود في التعليمة .

I قيمة فورية موجودة في التعليمة .
CO عدّاد البرنامج (Program counter) .
S عنوان رمزي ، تعبير قابل للتحويل :

$$S = D_2 + X_2 + B_2$$

أو $S = D_2 + B_2$

(S) مضمون العنوان S .
→ رمز للتخصيص ، أي نسخ منطقة في أخرى دون تهديم المنطقة الأصلية . مثلاً : $R_1 \rightarrow (S)$ يعني نسخ مضمون المرصف R_1 في المنطقة من الذاكرة بالعنوان S . لن نستعمل أبداً الترميز (R) للإشارة إلى مضمون المرصف R لأنه لن يوجد أي إبهام ، في حالة المرصف يتعلّق ذلك دائماً بالمضمون بينما يجب التمييز بين الإسم S للذاكرة ومضمونها .
((S)) من الممكن استعمال هذا التعبير للإشارة إلى أن مضمون العنوان S هو نفسه المعتمد كعنوان نأخذ منه المضمون .
CC يعني كود ـ الشرط .
الدلائل (indices) الدلائل 1 ، 2 ، 3 ترجع إلى الحقول المرتبطة بالتعليمة الآلية (فقرة 1.5) .

R_1.(24-31) تعني البتات 24 إلى 31 من المرصف رقم R_1 .
R_1 , R_{1+1} تعني المرصف المزدوج المؤلف من المراصف ذات الرقم R_1 وR_{1+1} . R_1 يكون رقماً مزدوجاً .

العناوين (adresses) نشير إلى أن العناوين تعني البايتة من اليسار لمنطقة ما ، وإن البتات من الكلمة ، من مرصف . . . هي مرقّمة من اليسار إلى اليمين إنطلاقاً من 0 .

(370) تشير إلى أن التعليمة غير موجودة إلا على المكنة 370 :

2.10 . كود العمليات الحرفية التذكيرية

كتابة كود ـ العمليات الرمزية يخضع إلى قواعد من المفيد الإشارة لها هنا . إنّ كود العملية يترجم الفعل المطلوب إجراؤه . السمة الأولى (أحياناً السمتان الأوليان) هي بداية الفعل الذي يُعبّر عن العمل .

مثلاً :

جمع	Add	A
شحن	LOAD	L

ST	STore	خزن
MVC	MoVe	نقل

الأحرف التالية هي معدّلات (1) أو أنّها تُميِّـز نوع المعطيات المُعالَجة (2) أو أيضاً النسق RR أو SI للتعليمات (3) .

أمثلة :

جمع منطقي Add Logicial	AL	(1)
تحويل إلى ثنائي ConVert Character	CVB	(2)
جمع معطيات من نوع بفاصلة متحركة قصير Add données de type E (flottant court)	AE	(2)
نقل السمات MoVe Characters	MVC	(2)
جمع معطيات من نوع D	AD	(2)
شحن بنسق RR	LR	(3)
شحن إيجابي بنسق RR	LPR	(3)
شحن مباشر بنسق SI	MVI	(3)

11 . الحساب بفاصلة ثابتة والحركات

1.11 . تعليمات الشحن والتخزين في المراصف العامة

هذه هي التعليمات التي تنسخ المتأثر في أحد المراصف :

« عنوان المتأثر , رقم المرصف LOAD »

وتنسخ مضمون المرصف في الذاكرة على عنوان معيّن :

« عنوان , رقم المرصف STORE »

هذه العمليات لا تؤثّر على المتأثر الأساسي . بعض التعليمات تؤدي إلى تركيز كود ـ الشرط CC ، لموقعين ثنائيين ينتميان إلى PSW (فصل 4) ، تبعاً لإشارة المتأثر المنقول حسب الإتفاق التالي :

بعد العملية فإن CC سيُركّز على[1] :

ـ 0 إذا كانت النتيجة صفراً .

ـ 1 إذا كانت النتيجة سلبية .

ـ 2 إذا كانت النتيجة إيجابية .

ـ 3 إذا كان هناك زيادة عن السعة (overflow) .

الزيادة عن السعة تؤدي عادة إلى إنقطاع في تنفيذ البرنامج . أي أنّه سيحدث خطأ يُعالجه نظام التشغيل . يوجد برنامج ، يُدعى برنامج إنقطاع «fixed point overflow» ، يعطي العلاج للمستعمل ويُوقف العمل في تنفيذ البرنامج بنهاية غير طبيعية . بإمكان المبرمج أن يقوم بتقنيع عملية الإنقطاع هذه في بعض الحالات بتركيز البتات المناسبة لقناع البرنامج في PSW .

وسندرس هذا الأمر لاحقاً (الفصل 19) .

---

(1) هذا الاتفاق هو صالح فقط للتعليمتين LOAD وSTORE وبعض التعليمات الأخرى . وسنرى كيف يتمّ تركيز CC لكلّ مجموعة تعليمات .

المتأثر 1 هو دائماً مرصف ، والمتأثر الثاني يُمكن أن يكون مرصفاً ، نصف كلمة أو كلمة ـ ذاكرة .

من المهم أن نشير إلى أن المتأثرات الموجودة على العناوين المشار إليها بواسطة S يجب أن تُحصر في حدود كلمات أو نصف ـ كلمات حسب التعليمات .

LR	R_1,R_2	RR	COP=18	LOAD $R_2 \to R_1$
L	$R_1,D_2(X_2,B_2)$	RX	COP=58	LOAD $(S_2) \to R_1$

CC لا يتغيّر

LH	$R_1,D_2(X_2,B_2)$	RX	COP=48	LOAD HALFWORD $(S_2) \to R_1$

يُعتبر المتأثر الثاني كعدد صحيح بإشارة وبطول 16 بتة . يُوسَّع إلى 32 بتة قبل التحويل. CC لا يتأثر .

LCR	R_1,R_2	RR	COP=13	LOAD COMPLEMENT $R_2 \to R_1$

يخزِّن عكس(مكمِّل إلى 2) R_2 في R_1 . overflow إذا أكملنا العدد السلبي الأقصى . يوضع CC حسب الإشارة النهائية لِـ R_1 .

LPR	R_1,R_2	RR	COP=10	LOAD POSITIVE القيمة المطلقة لـ $R_2 \to R_1$

سيحدث زيادة عن السعة (overflow) إذ أكملنا العدد السلبي الأقصى . يُركّز CC على 0 ، 2 أو 3 حسب النتيجة .

LNR	R_1,R_2	RR	COP=11	LOAD NEGATIVE

المكمل إلى 2 للقيمة المطلقة لِـ R_2 يُخزَّن في R_1 . لن يحدث overflow . CC يُركّز على 0 أو 1 .

LTR	R_1,R_2	RR	COP=12	LOAD AND TEST $R_2 \to R_1$

تعليمة شبيهة بـ LR باستثناء كون الإشارة النهائية لـ R_1 تُركّز CC . R_1 يمكن أن يكون معادلاً لـ R_2 .

LM	$R_1,R_3,D_2(B_2)$	RS	COP=98	LOAD MULTIPLE

المواقع المتتالية للذاكرة ، انطلاقاً من العنوان S_2 ستشحن في المراصف العامة R_1 ، R_{1+1} ، . . . ، R_3 . في هذه التعليمة يُفترض بأن يتبع المرصف 0 المرصف 15 . هكذا :

LM 15, 1, ALPHA

ستشحن الكلمة ذات العنوان ALPHA في المرصف 15 ، وتلك ذات العنوان ALPHA+4 في المرصف 0 وهكذا دواليك . تُستخدم هذه التعليمة بشكلٍ خاص لترميم إطار البرنامج

CC لا يتأثر .

LA $R_1,D_2(X_2,B_2)$ RX COP=41 LOAD ADDRESS

$S_2 \rightarrow R_{1(8-31)}$ $0 \rightarrow R_{1(0-7)}$

تخزَّن القيمة ذات العنوان S_2 في البتات من 8 إلى 31 من المرصف R_1 . يتمّ تصفير البتات من 0 إلى 7 . وتنطبق هنا قواعد حساب العنوان ، أي أن القيمة $D_2+ X_2 + B_2$ تُخزَّن (عنوان فعلي) . من الممكن أن تأخذ نفس المرصف لـ X_2, R_1 أو B_2 . المرصف 0 لا يؤخذ أبداً وكأنه قاعدة أو مرصف تأشير .

الاستعمال : أنظر التمارين

- شحن عنوان في مرصف ،
- شحن عدد غير سلبي أصغر أو يعادل 4095 (القيمة القصوى للإزاحة) في مرصف ،
- زيادة مضمون مرصف بقيم أصغر أو تساوي 4095 .

IC $R_1,D_2(X_2,B_2)$ RX COP=43 INSERT CHARACTER

$(S_2) \rightarrow R_{1(24-31)}$

لا يتغيَّر $R_{1(0-23)}$

يتم تخزين بايتة واحدة بعنوان S_2 في R_1 . CC لا يتأثر .

ICM $R_1,M_3,D_2(B_2)$ RS COP=BF INSERT CHARACTERS UNDER MASK
(370)

تُربط البتات الأربع من القناع M_3 بالبتات الأربع للمرصف R_1 . البايتات من R_1 المرتبطة بالبتات «1» من القناع يتم شحنها مع البايتات المتتالية من S_2 . طول المتأثر الثاني يعادل عدد «1» في القناع .

يُركَّز كود الشرط :

CC = 0 : جميع البتات الداخلة هي مصفَّرة أو القناع مصفَّر ،

CC = 1 : البتة ذات الوزن الأكبر في S_2 هي «1» ،

CC = 2 : البتات ذات الوزن الأكبر في S_2 هي «0» ولكن جميع البتات الداخلة ليست صفراً .

وفي الختام فإن CC يُركَّز حسب إشارة S_2 .

ST $R_1,D_2(X_2,B_2)$ RX COP=50 STORE

$R_1 \rightarrow (S_2)$

CC والمرصف R_1 يبقيان بدون تعديل .

STH $R_1,D_2(X_2,B_2)$ RX COP=40 STORE HALFWORD

$R_{1(16-31)} \rightarrow (S_2)$

المتأثر الثاني هو بطول 2 بايتة . CC يبقى بدون تعديل .

STM $R_1,R_3,D_2(B_2)$ RS COP=90 STORE MULTIPLE

المراصف العامة من R_1 إلى R_3 يتم تخزينها في مواقع متتالية من الذاكرة بدءاً من العنوان S_2 . الرقم 0 للمرصف 0 مُفترض أنّه يتبع الرقم 15 بشكل يؤدي معه تنفيذ التعليمة ST 15, 1, ALPHA إلى تخزين

المراصف 15 ، 0 ، 1 بالعناوين ALPHA ، ALPHA+4 ، . . .
تستخدم التعليمة بشكل خاص لحفظ إطار البرنامج . CC يبقى بدون تغيير .

STC $R_1,D_2(X_2,B_2)$ RX COP=42 STORE CHARACTER
$R_{1(24\text{-}31)} \rightarrow (S_2)$
R_1 وCC يبقيان بدون تعديل .

STCM $R_1,M_3,D_2(B_2)$ RS COP=BE STORE CHARACTERS UNDER MASK
(370)
البتات الأربع من القناع M_3 ترتبط بالأربع بايتات من المرصف R_1 .
أما بايتات R_1 ، والمختارة بوجود «1» في القناع ، فيتم تخزينها بشكل متراص على العنوان S_2 . كود الشرط CC لا يتغيّـر .

2.11 . التعليمات الحسابية بفاصلة ثابتة

هي التعليمات التي تعمل على معطيات ممثلة بفاصلة ثابتة . تكوَّد القيم السلبية بواسطة المكمِّل إلى 2 . كما تقوم بالعمليات الأربع الأساسية بين مرصف ومرصف أو بين مرصف وذاكرة . الضرب والجمع يستعملان مراصف مزدوجة (فقرة 1.10) . هذه التعليمات تؤدي إلى تعديل CC حسب إشارة النتيجة ، وحسب الإتفاق الجاري كما في 1.11 .

CC = 0 إذا كانت النتيجة صفراً .
CC = 1 إذا كانت النتيجة سلبية .
CC = 2 إذا كانت النتيجة إيجابية .
CC = 3 إذا كان هناك overflow .

يمكن قطع التعليمة في حالة حدوث حادثة غير طبيعية ، كما يلي :
- عنوان من خارج المنطقة المخصَّصة .
- جبهة متأثر غير صحيحة ، مرصف مزدوج معني بشكل سيء .
- فيض عن السعة overflow .

AR R_1,R_2 RR COP=1A ADD
$R_1+R_2 \rightarrow R_1$
A $R_1,D_2(X_2,B_2)$ RX COP=5A ADD
$R_1+(S_2) \rightarrow R_1$

لا يتغيّـر المتأثر الثاني . يتم تركيز كود الشرط CC ، إحتمال حصول overflow .

AH $R_1,D_2(X_2,B_2)$ RX COP=4A ADD HALFWORD

$(S_2)+R_1 \rightarrow R_1$

المتأثر (S_2) هو على نصف كلمة . يُوسَّع الى كلمة قبل العملية . يتم تركيز CC .
احتمال حصول Overflow .

SR R_1,R_2 RR COP=1B SUBTRACT

$R_1 - R_2 \rightarrow R_1$

S $R_1,D_2(X_2,B_2)$ RX COP=5B SUBTRACT

$R_1 - (S_2) \rightarrow R_1$

المتأثر الثاني لا يتعدل يتم تركيز CC .

SH $R_1,D_2(X_2,B_2)$ RX COP=4B SUBTRACT HALFWORD

$R_1 - (S_2) \rightarrow R_1$

المتأثر S_2 هو على نصف كلمة ، يوسَّع إلى 32 بتة قبل العملية . يتم تركيز CC .

MR R_1,R_2 RR COP=1C MULTIPLY

$R_{1+1} \times R_2 \rightarrow R_1, R_{1+1}$

M $R_1,D_2(X_2,B_2)$ RX COP=5C MULTIPLY

$R_{1+1} \times (S_2) \rightarrow R_1, R_{1+1}$

المرصف R_1 المذكور في التعليمة يجب أن يكون مرصفاً مزدوجاً . المتأثر الأول يجب أن يكون موجوداً في R_{1+1} ومحصوراً لجهة الشمال . النتيجة ستوضع في R_1 ، R_{1+1} . لا احتمال لحدوث overflow ، لا يتم تركيز CC .

MH $R_1,D_2(X_2,B_2)$ RX COP=4C MULTIPLY HALFWORD

$R_{1+1} \times (S_2) \rightarrow R_1,R_{1+1}$

المرصف R_1 يجب أن يكون مرصفاً مزدوجاً ؛ S_2 يتألف من 16 بتة ويُعتبر كعدد صحيح بإشارة يُوسَّع إلى 32 بتة قبل العملية . لا يحدث overflow ولا يتم تركيز CC .

DR R_1,R_2 RR COP=1D DIVIDE

$R_1,R_{1+1} : R_2$ → R_1 باقي / → R_{1+1} قيمة القسمة

R_1 هو مرصف مزدوج . يتمتع الباقي بنفس إشارة المقسوم . عندما لا تسع 32 بتة نتيجة القسمة يحدث overflow . لا يتم تركيز CC .

D $R_1,D_2(X_2,B_2)$ RX COP=5D DIVIDE

$R_1,R_{1+1} : (S_2)$ → R_1 reste / → R_{1+1} quotient

R_1 يجب أن يكون مرصفاً مزدوجاً . للباقي نفس إشارة المقسوم . عندما لا تسع 32 بتة نتيجة القسمة يكون هناك فيض عن السعة . لا يتمّ تركيز CC .

ملاحظات :

دراسة هذه التعليمات تسمح لنا بملاحظة إن النتيجة تحلّ دائماً مكان المتأثر الأول الذي يضيع منا . بينما لا يتم تعديل المتأثر الثاني . التعليمات التي تجري على نصف كلمة تفترض توسيع نصف الكلمة إلى كلمة قبل العملية .

3.11 . عمليات المقارنة بفاصلة ثابتة

تؤثر تعليمات المقارنة فقط على مضمون كود الشرط . هذه التعليمات هي خاصة حسب نوع تمثيل المعطيات المُقارنة . سندرس هنا تلك المتعلقة بالفاصلة الثابتة . كما في التعليمات التي رأيناها، فإن المتأثر الأول هو دائماً موجود في مرصف معين والمتأثر الثاني في مرصف آخر أو في الذاكرة . يجري تركيز CC حسب الطريقة التالية :

CC = 0 إذا كان المتأثر الأول = المتأثر الثاني .
CC = 1 إذا كان المتأثر الأول أصغر من المتأثر الثاني .
CC = 2 إذا كان المتأثر الأول أكبر من المتأثر الثاني .
CC = 3 لا يُستعمل .

CR	R_1,R_2	RR	COP=19	COMPARE
C	$R_1,D_2(X_2,B_2)$	RX	COP=59	COMPARE

المقارنة هي جبرية وتتعلّق بـ 32 بتة . يتم تركيز مضمون CC .

CH	$R_1,D_2(X_2,B_2)$	RX	COP=49	COMPARE HALFWORD

يُوسّع المتأثر الثاني إلى 32 بتة قبل المقارنة مع إنتشار بتة الإشارة.
يتم تركيز CC .

4.11 الجمع والطرح المنطقي

نعني بالجمع والطرح المنطقي، تعليمات تعدِّل مضمون CC بطريقة مختلفة عن الجمع والطرح العادي الذي رأيناه أعلاه . إضافة لذلك فإن overflow لا يؤدي إلى قطع البرنامج

يتم تركيز CC على الشكل التالي :
CC = 0 إذا كانت النتيجة صفراً بدون مرحّل .
CC = 1 إذا كانت النتيجة مختلفة عن 0 بدون مرحّل (no carry)
CC = 2 إذا كانت النتيجة صفراً مع مرحّل .
CC = 3 إذا كانت النتيجة مختلفة عن صفر مع مرحّل .

ALR R_1,R_2	RR	COP=1E	ADD LOGICAL $R_2 + R_1 \rightarrow R_1$
AL $R_1,D_2(X_2,B_2)$	RX	COP=5E	ADD LOGICAL $(S_2) + R_1 \rightarrow R_1$

SLR R_1,R_2	RR	COP=1F	SUBTRACT LOGICAL $R_1 - R_2 \rightarrow R_1$
SL $R_1,D_2(X_2,B_2)$	RX	COP=5F	SUBTRACT LOGICAL $R_1 - (S_2) \rightarrow R_1$

5.11 . التحريك من الذاكرة إلى الذاكرة

تتم في أغلب الأحيان بواسطة تعليمات من نوع SS . لا يوجد أي تقييد فيما يتعلّق بالاصطفاف (alignement) . يمكن أن يتم تركيز الطول بشكل واضح في التعليمة : MVC ZONE 1 (L), ZONE 2 أو ضمنياً MVC ZONE 1, ZONE 2 . يقوم عندها المؤول باختيار خاصية ـ الطول الخاصة بالمتأثر الأول L'ZONE 1 . الطول المؤول هو الطول المذكور في التعليمة ناقص 1 . يمكن للمتأثرين أن يتراكبا ، ونجد هذه الميزة مستعملة في التمرين 6.11 .

MVI $D_1(B_1),I_2$	SI	COP=92	MOVE $I_2 \rightarrow (S_1)$

يتم تخزين البايتة المباشرة I_2 في S_1 .

MVC $D_1(L,B_1),D_2(B_2)$	SS	COP=D2	MOVE

$(S_2) \rightarrow (S_1)$ بطول L .

الحركة تتم من اليسار إلى اليمين . العملية هي غير قابلة للانقطاع عند نقل بايتتين . يسمح بالتراكب وفي هذه الحالة يجدر الانتباه إلى أنّ الحركة تجري من اليسار إلى اليمين من أجل الحصول على النتيجة .

MVCL R_1,R_2 (370)	RR	COP=0E	MOVE LONG

نسخ المتأثر الثاني في المتأثر الأول .

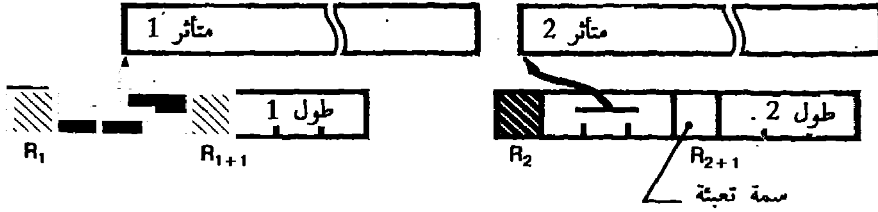
R_1 (8 – 31) يحتوي على عنوان المتأثر الأول ،

R_{1+1} (8 – 31) طول المتأثر الأول ،

R_2 (8 – 31) عنوان المتأثر الثاني ،

R_{2+1} (0 – 7) سمة تعبئة ،

R_2 + (8 – 31) طول المتأثر الثاني .



الحركة تتم من اليسار إلى اليمين ، لكل باية على حدة . التعليمة هي قابلة للانقطاع عند نسخ بايتين . إذا كان طول المتأثر الثاني هو أصغر من طول المتأثر الأول ، يتم تكملة المتأثر الأول بسمة تعبئة . يمكن تراكب المناطق بشرط أن لا يقوم النسخ بتعديل باية جرى تعديلها سابقاً .

يجري تركيز CC على الشكل التالي :

CC = 0 إذا كان كلا المتأثرين بنفس الطول ،

CC = 1 المتأثر الأول هو أقصر ،

CC = 2 المتأثر الأول هو أطول ،

CC = 3 إذا أدت عملية التطابق إلى تعديل في باية معدَّلة أصلاً .

يُمكن إستعمال هذه التعليمة لتصفير الذاكرة .

MVN $D_1(L,B_1),D_2(B_2)$ SS COP=D1 MOVE NUMERIC

نسخ نصف ـ بايتات بالوزن الأضعف من (S2) في أنصاف ـ بايتات الوزن الأضعف من (S1) . تبقى أنصاف ـ البايتات بالوزن الأقوى دون تعديل . يسمح بالتراكب وبهذا الصدد نعطي الملاحظة نفسها كما بالنسبة لـ MVC

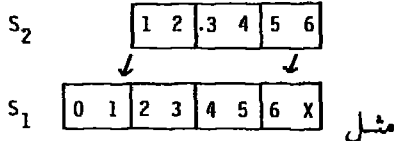
MVZ $D_1(L,B_1),D_2(B_2)$ SS COP=D3 MOVE ZONES

نسخ نصف بايتات بالوزن الأقوى من (S2) في نصف بايتات الوزن الأقوى من (S1) . تبقى أنصاف ـ البايتات بالوزن الأضعف دون تعديل . يسمح بتراكب الحيِّزات وبهذا الصدد نعطي الملاحظة نفسها كما بالنسبة لـ MVC .

MVO $D_1(L_1,B_1),D_2(L_2,B_2)$ SS COP=F1 MOVE WITH OFFSET

نسخ من (S2) في (S1) مع إزاحة إلى اليسار مقدار نصف باية . العملية تتم من اليمين إلى اليسار ، باية بعد باية . لا يتم تغيير آخر باية لجهة اليمين .

مثل



تمارين

تمرين 1.11 ـ ضع في الصفر الثنائي أحد المراصف (أعطِ حلّين لتعليمة واحدة دون حجز ثوابت) .

تمرين 2.11 ـ غيِّر إشارة المرصف (تمثيل ثنائي) .

تمرين 3.11 . ضع جميع بتات المرصف في 1 .

تمرين 4.11 ـ اشحن القيمة 2048 في مرصف ، ثمّ القيمة 4095 (دون حجز ثابتة) بعد ذلك اشحن 4096 .

تمرين 5.11 ـ زد مضمون أحد المراصف مقدار 4 .

تمرين 6.11 ـ عبِّىء منطقة بطول L≥ 256 باية بنجوم (تعليمتان) .

12 التفريعات

نفهم بالتفريع كل تعديل في مضمون عداد البرنامج يؤدي إلى إنقطاع في الدوران المتتالي للتعليمات .

عوّدتنا دراسة اللغات المتطورة على اعتبار نوعين من الإنقطاعات في المتتالية :

- الإنقطاعات الإلزامية (GOTO في لغة فورتران) .
- الإنقطاعات المشروطة (IF) .

في لغة المؤوّل ، فإن الإنقطاعات المشروطة تنتج إمّا عن اختيار لقيمة مأخوذة من كود الشرط ، إمّا عن اختبار لقيمة مأخوذة من مرصف عام . التعليمتان BC وBCR تفحصان كود الشرط CC والتعليمات BCT ، BCTR ، BXH ، BXLE تُخفِّض أو تزيد من مضمون مرصف وبعد ذلك تفحص قيمته .

يمكن تنفيذ الإنقطاعات الإلزامية بواسطة BC وBCR .

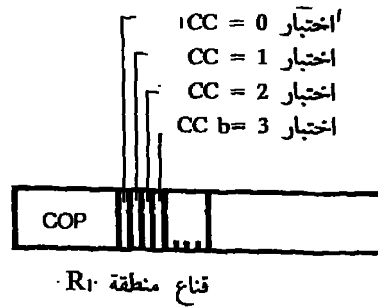
1.12 . الكود - الشرط

لقد التقيناه عند دراسة التعليمات السابقة . ونذكّر بأنّه عبارة عن مؤشر بموقعين ثنائيين ، ينتميان إلى PSW (البتتان، 34 ، 35) ويركّزان بواسطة بضع تعليمات حسب النتيجة الحاصلة . التعليمات الحسابية ، مثلاً ، التركيز حسب إشارة النتيجة ، تعليمات المقارنة حسب القيمة النسبية لمتأثرين .

الكود الشرطي CC يمكن أن يأخذ إذن أربع قيم ثنائية 00 ، 01 ، 10 ، 11 يتم مراجعتها في التعليمات بواسطة 0 ، 1 ، 2 ، 3 .

2.12 . التعليمات التي تفحص الكود الشرطي (CC) : BCR وBC

هذه التعليمات تستعمل المنطقة R1 ، المكوّنة من أربع بتات ثنائية ، من نسقها الآلي ، ليس كرقم مرصف بل كقناع : كل بتة تعادل 1 وموجودة في هذه المنطقة تناسب إختبار إحدى القيم الأربع التي نحصل عليها بواسطة CC حسب الإتفاق التالي :



قناع منطقة R_1

هكذا ، فالقناع المُعادل 1100 (ثنائياً) سيسمح باختبار الشروط CC=0 أو CC=1 . الشرط المُختار فعلاً يتعلّـق إذاً بالتعليمة التي أدت إلى تركيز CC .

لقد رأينا أن CC تركّـز حسب الطريقة التالية :

كود الشرط	0	1	2	3
تعليمات حسابية نتيجة	=0	<0	>0	فيض عن السعة
تعليمات مقارنة ؛ متأثر أول	=	<2e	>2e	---

القناع المُعادل لـ 1100 (أي C بالنظام السادس عشري أو 12 بالعشري) يناسب الاختبارات التالية :

- نتيجة سلبية أو صفر بعد تعليمة حسابية .
- متأثر أول أصغر من المتأثر الثاني بعد تعليمة مقارنة .

BCR M_1,R_2 RR COP=07 BRANCH ON CONDITION

M_1 هي القناع المذكور أعلاه .

بعد تنفيذ الشرط ، هناك تفريع إلى العنوان المخزّن في R_2 . وإلا سيتابع التنفيذ بالتوالي . مما يترجَم على الشكل التالي : الشرط المنفذ. $R_2 \rightarrow CO$

وإلا $CO + 2 \rightarrow CO$

BC $M_1,D_2(X_2,B_2)$ RX COP=47 BRANCH ON CONDITION

M_1 قناع .

إذا تم تنفيذ الشرط فسيحدث تفريع إلى العنوان $D_2 + X_2 + B_2$
وإلا فإن التنفيذ سيتتابع بالتوالي ، مما يترجم على الشكل التالي :
في حال تنفيذ الشرط : $D_2 + X_2 + B_2 \rightarrow CO$
وإلا $CO + 4 \rightarrow CO$
$D_2 + X_2 + B_2$ عنوان التفريع ..

في لغة المؤول ، يحدد القناع M_1 بواسطة تعبير مطلق ، عادة رقم عشري .

BC 15, ALPHA أو BCR 15,R يناسبان القناع 1111 . يتعلّق ذلك إذاً بالتفريع المنتظم لأنه مهما تكن قيمة CC هناك تفريع .

BCR 0,R أو BC 0,ALPHA هي عبارة عن تعليمات دون فعل لأنه لن يتم إختبار أي شرط . وهي تتميّز بأنها بدون فعل .

الأكواد الحرفية التذكيرية الموسّعة

وفي النهاية كي يتم تفادي تحديد القناع الخاص ولتذكر الإتفاقات المذكورة أعلاه ، فإن المؤول يسمح باستعمال كود حرفي حسب الشرط المفحوص .

ويقوم بمهمة ترجمة الكود الحرفي إلى BC أو BCR .

هكذا :

$B \quad D_2(X_2,B_2)$ يُناسب تفريعاً غير شرطي
BC 15, D_2 (X_2, B_2)

$BR \quad R_2$ يناسب تفريعاً غير شرطي
BCR 15,R_2

BNE $D_2(X_2,B_2)$ يناسب تفريعاً معيّناً وإلا يعادل
BC 7,D_2 (X_2, B_2)

سنجد في الملحق اللائحة الكاملة للكود الحرفي التذكيري الموسّع . سنلاحظ إن الأكواد الحرفية تتعلّق بالتعليمة التي تقوم بتركيز الكود الشرطي . من المفيد ، لوضوح البرنامج ، إستعمال هذه الأكواد الحرفية التذكيرية . ونركّز على كون هذه الأكواد العملية لا تُناسب سوى مع 2 كود ـ مكنة . ونشير ، كما ذكرنا في الفقرة 2.10 ، الى أن الأكواد التي تنتهي بـ R تناسب تعليمات بنسق RR أو BCR .

3.12 . التعليمات التي تفحص القيمة المأخوذة من مرصف (مؤشر)

أربع تعليمات BCT ، BCTR ، BXH وBXLE تسمح بتعديل مضمون

المرصف والتفريع إلى عنوان معيّن عندما تصبح قيمته معادلة ، أقل أو أكبر من كمية محدّدة .

BCTR R_1,R_2 RR COP=06 BRANCH ON COUNT
$R_1 - 1 \rightarrow R_1$

إذا كانت $R_1 \neq 0$: $R_2 \rightarrow CO$ (تفريع إلى العنوان الموجود في R_2) .
وإلا $CO + 2 \rightarrow CO$ (تنفيذ التعليمة التالية) .
ملاحظة : إذا كان R_2 هو المرصف 0 فالعدّ يتم بدون تفريع .

BCT $R_1,D_2(X_2,B_2)$ RX COP=46 BRANCH ON COUNT
$R_1 - 1 \rightarrow R_1$

إذا : $R_1 \neq 0$: $S_2 \rightarrow CO$ (تفريع إلى العنوان S_2)
وإلا : $CO + 4 \rightarrow CO$ (تنفيذ التعليمة التالية) .

BXH $R_1,R_3,D_2(B_2)$ RS COP=86 BRANCH ON INDEX HIGH

1 ـ زيادة مضمون R_1 : $R_1 + R_3 \rightarrow R_1$
2 ـ عندما تصبح R_1 أكبر من المرجعية : تفريع . المرجعية هي R_3
R_{3+1}

أ ـ R_3 هو مرصف برقم مفرد .
R_3 هو مرجع المقارنة والزيادة .

فإذاً : $R_1 + R_3 \rightarrow R_1$ بعد ذلك ، إذا كان $R_1 > R_3$
عندئذٍ $S_2 \rightarrow CO$ (تفريع إلى S_2)
وإلاّ $CO + 4 \rightarrow CO$ (متابعة على التوالي) .

ب ـ R_3 هو مرصف برقم مزدوج
نستعمل المرصف المزدوج R_3 وR_{3+1}
R_3 هو الزيادة وR_{3+1} هو المرجعية . إذن $R_1 + R_3 \rightarrow R_1$ ثمّ
إذا كان $R_1 > R_{3+1}$ عندئذٍ $S_2 \rightarrow CO$ (التفريع إلى S_2) .
وإلاّ $CO + 4 \rightarrow CO$ (متابعة المتالية) .

ملاحظة :
يجب أن لا نخلط هنا بين المصطلح إشارة مع مرصف المؤشر للتعليمات RX .
المقارنة تتم جبرياً . ويتم إهمال overflow عند الجمع .

BXLE $R_1,R_3,D_2(B_2)$ RS COP=87 BRANCH ON INDEX LOW OR EQUAL

1 ـ زيادة R_1 : $R_1 + R_3 \rightarrow R_1$

2 ـ عندما يصبح R_1 أصغر أو يعادل المرجعية : تفريع المرجعية . المرجعية
• هي R_3 أو R_{3+1} .

أ ـ R_3 هو مرصف برقم مفرد .
R_3 هو مرجعية المقارنة والزيادة .
فإذاً : $R_1 + R_3 \rightarrow R_1$ بعد ذلك ، إذا كان $R_1 \leqslant R_3$ عندئذٍ
$S_2 \rightarrow CO$ (تفريع إلى S_2) وإلا $CO + 4 \rightarrow CO$ (متابعة
المتتالية)

ب ـ R_3 هو مرصف برقم مزدوج .
R_3 هو الزيادة ، R_{3+1} هو المرجعية .

فإذاً $R_1 + R_3 \rightarrow R_1$ ثمّ إذا كان $R_1 \leqslant R_{3+1}$ عندئذٍ $S_2 \rightarrow CO$
(تفريع إلى S_2) وإلا $CO + 4 \rightarrow CO$ (متابعة المتتالية) .

ملاحظة : يجب أن لا نخلط هنا بين المصطلح مؤشر مع مرصف المؤشر للتعليمات RX . تتمّ المقارنة جبرياً . يتم إهمال overflow عند الجمع .

4.12 . تفريع مع عودة

مشكلة التفريع مع تخزين عنوان التعليمة التي تلي تعليمة التفريع تحدث عند دعوة برنامج ثانوي . هناك تعليمتان BALR وBAL موجّهتان لهذا الإستعمال .

BALR R_1,R_2 RR COP=05 BRANCH AND LINK
$CO \rightarrow R_{1(8\text{-}31)}$ (تخزين عنوان العودة)
$CC \rightarrow R_{1(0\text{-}7)}$
$R_{2(8\text{-}31)} \rightarrow CO$ (تفريع)

ملاحظة :

نذكّر بأنّ قيمة عداد البرنامج CO تتغيّر خلال تنفيذ التعليمة . هكذا ، فعنوان التعليمة التالية حسب BALR هو المخزّن في R_1 . BALR R_1,0 يقوم بتخزين العنوان التالي في R_1 ولكن لا تفريع . هناك إذن تتابع للمتتالية . هذا الشكل هو الأكثر استعمالاً لشحن مرصف قاعدي بالقيمة التالية لعداد البرنامج .

إذا كانت التعليمة BALR موجودة على العنوان 50000 ، فإن القيمة 50002 ستخزّن في R_1 .

BAL $R_1,D_2(X_2,B_2)$ RX COP=45 BRANCH AND LINK
$CO \rightarrow R_{1(8\text{-}31)}$ (تخزين عنوان العودة)
$CC \rightarrow R_{1(0\text{-}7)}$
$S_2 \rightarrow CO$ (تفريع إلى العنوان S_2)

كما في BALR ، فعنوان التعليمة التالية سيخزّن في R_1 . إذا كانت BAL موجودة على العنوان 50000 فإن مضمون R_1 هو 50004 .

EX $R_1,D_2(X_2,B_2)$ RX   COP=44   EXECUTE

هذه التعليمة تسمح بتنفيذ تعليمة واحدة موجودة خارج التتابع الطبيعي للعنوان S_2 . بعد ذلك ، فإن العمل يُعاود بالتوالي .

يتم تنفيذ عملية « أو » متضمّنة بين البتات $R_{1(24-31)}$ و $S_{2(8-15)}$) تسمح بتعديل هذا الحقل من التعليمة (رقم المرصف ، قيمة تلقائية أو طول) .

إذا كان R_1 هو المرصف 0 فلا يتم تنفيذ العملية « أو » (OR) . كما لا يمكن تنفيذ عملية التحويل .

تطبيق

عندما نرغب بإجراء نقل للمعلومات MVC من منطقة لا نعرف طولها إلا في لحظة التنفيذ ـ هذه الحالة تحدث عند معالجة التسجيلات بطول مُتغيِّر ، يكون طول الفقرة موجوداً في رأسها ـ من الممكن إذاً تنفيذ التعليمة «MVC» . والطريقة هي التالية : شحن الطول في $R_{1(24-31)}$:

```
BCTR   R1, 0                    ( تنقيص 1 )
EX     R1,MOVE
--     --------
MOVE   MVC      --------
```

تُنفَّذ MVC مع الطول المطلوب دون أن يكون هناك تعديل للتعليمة في الذاكرة . التعليمة MVC لا تتعدَّل إلا خلال مدة التنفيذ . ويمكن أن تكون موجودة في أي مكان ولكن يُفضَّل أن تكون EX وMVC موجودتين في نفس الصفحة من الذاكرة كي لا نقع في خطأ محتمل في نقص الصفحة .

تمارين :

تمرين 1.12 . أكتب متتالية التعليمات التي تسمح بتكرار N مرّة إحدى عمليات المعالجة .

تمرين 2.12 . إحسب مجموع عناصر جدول من الكلمات يحتوي على أعداد بفاصلة ثابتة .

تمرين 3.12 . إعكس سلسلة من السمات CH1 في CH2 .

تمرين 4.12 . نقّص مضمون المرصف 1 (تعليمة واحدة) .

تمرين 5.12 . إشحن مرصفاً معيّناً بالعنوان الجاري زائد 2 .

13 . العمليات المنطقية

1.13 . الدوال المنطقية

يسمح الكومبيوتر IBM 360/370 بعنونة البايتة ، ومن غير الممكن الإشارة إلى بتة معيّـنة داخل البايتة . ولكن بسبب وجود تعليمات الإزاحة (Shift) والتعليمات المنطقية سيكون بإمكاننا إختبار أو تعديل مضمون إحدى البتات من داخل الكلمة .

العمليات المنطقية الموجودة هي « و » (AND) ، الجمع « أو » (OR) و« أو المقتصرة » (EOR) . جدول العمليات المنطقية هو التالي :

A	1	0	1	0	تعليمات
B	1	1	0	0	
A AND B	1	0	0	0	NR N NI NC
A OR B	1	1	1	0	OR O OI OC
A FOR B	0	1	1	0	XR X XI XC

2.13 . التعليمات المنطقية

المتأثرات هي :

- مرصفان عامّـان (شكل RR) : التعليمات NR ، OR ، XR ،
- مرصف وكلمة - ذاكرة (شكل RX) : التعليمات N ، O ، X ،
- بايتة موجودة في التعليمة وبايتة موجودة في الذاكرة (الشكل SI عنونة مباشرة) : التعليمات NI ، OI ، XI ،
- سلسلتان من البايتات في الذاكرة (شكل SS) : التعليمات NC ، OC ، IC .

تُوضع النتيجة دائماً في المتأثـر 1ـ .

يتم تركيز كود الشرط حسب الطريقة التالية :

CC	
0	إذا كانت النتيجة تعادل صفر
1	إذا كانت النتيجة مختلفة عن صفر

عمليات الإنقطاع الممكنة تتعلّق ، كالعادة ، بمسألة العنونة : تعدٌّ على المنطقة المخصّصة من الذاكرة ، تعدٍّ على المنطقة الممكنة من الذاكرة أو مشكلة الزيادة في مضمون المراصف المزدوجة .

التقاطع « و » (AND)

NR R_1,R_2 RR COP=14 AND
R_1 «And» $R_2 \rightarrow R_1$
تتم العملية على 4 بايتات .

N $R_1,D_2(X_2,B_2)$ RX COP=54 AND
R_1 «And» $(S_2) \rightarrow R_1$
تتمّ العملية على أربع بايتات .

NI $D_1(B_1),I_2$ SI COP=94 AND
(S_1) «And» $I_2 \rightarrow (S_1)$
I_2 هي قيمة تلقائية موجودة في التعليمة . العملية تتم على بايتة واحدة .

NC $D_1(L,B_1),D_2(B_2)$ SS COP=D4 AND
(S_1) «And» $(S_2) \rightarrow (S_1)$
العملية تتم بين منطقتين من الذاكرة بطول مشترك هو L بايتة . وتجري العملية بايتة بعد بايتة من اليسار إلى اليمين . كل شيء يسير كما لو كانت كل بايتة محسوبة ومخزّنة في الذاكرة قبل العبور إلى البايتة التالية .

تطبيق عملي :

تصفير إحدى البتات .

الجمع « أو »

OR R_1,R_2 RR COP=16 OR
R_1 «OR» $R_2 \rightarrow R_1$
تتم على أربع بايتات

O $R_1,D_2(X_2,B_2)$ RX COP=56 OR
(S_2) «OR» $R_1 \rightarrow R_1$
تتمّ على أربع بايتات .

OI $D_1(B_1),I_2$ SI COP=96 OR
(S_1) « OU » $I_2 \rightarrow (S_1)$

I_2 هي قيمة موجودة في التعليمة . تجري العملية على بايتة واحدة .

OC $D_1(L,B_1),D_2(B_2)$ SS COP=D6 OR
(S_2) «OR» $(S_1) \rightarrow (S_1)$

تتمّ العملية على منطقتين من الذاكرة بطول مشترك هو L بايتة . وتتم بايتة بعد أخرى من اليسار إلى اليمين .

تطبيق عملي :

جعل إحدى البتات تعادل 1 .

« أو المقتصرة » (EOR)

XR R_1,R_2 RR COP=17 EXCLUSIVE OR
R_1 «EOR» $R_2 \rightarrow R_1$

تتم العملية على أربع بايتات .

X $R_1,D_2(X_2,B_2)$ RX COP=57 EXCLUSIVE OR
R_1 «EOR» $(S_2) \rightarrow R_1$

تتم العملية على أربع بايتات .

XI $D_1(B_1),I_2$ SI COP=97 EXCLUSIVE OR
(S_1) «EOR» $I_2 \rightarrow (S_1)$

I_2 هي قيمة تلقائية موجودة في التعليمة . تتم العملية على بايتة واحدة .

XC $D_1(L,B_1),D_2(B_2)$ SS COP=D7 EXCLUSIVE OR
(S_1) «EOR» $(S_2) \rightarrow (S_1)$

تجري العملية على منطقتين من الذاكرة بطول مشترك L بايتة ، وتجري بايتة بعد بايتة من اليسار إلى اليمين كما لو كانت كلّ بايتة قد جرى حسابها وتخزينها في الذاكرة قبل العبور إلى البايتة التالية .

تطبيق عملي :

عكس البتة ، مُكمِّل منطقي ، تصفير منطقة من الذاكرة .

3.13 . المقارنات المنطقية

كما في جميع العمليات المنطقية تجري معالجة جميع البتات بنفس الطريقة . لا وجود لأي تمييز للبتة ذات الوزن الأعلى . تتم المقارنة من اليسار إلى اليمين وتتوقف عند أوّل معادلة . يُركّز كود الشرط حسب الطريقة التالية :

(نذكّر أنّ المتأثّر الأول هو ذلك الذي يتم بلوغه في التعليمة بواسطة المؤشر 1 . الإنقطاعات الممكنة هي تلك المتعلقة بالعنونة وتلك المتعلقة بحدودالكلمات).

0	إذا كانت المتأثرات متساوية
1	إذا كان المتأثر الأول أصغر من المتأثر الثاني
2	إذا كان المتأثر الأول أكبر من المتأثر الثاني
3	غير مستعمل

CLR R_1,R_2 RR COP=15 COMPARE LOGICAL

مقارنة بين كامل المراصف .

CL $R_1,D_2(X_2,B_2)$ RX COP=55 COMPARE LOGICAL

مقارنة على أربع بايتات .

CLI $D_1(B_1),I_2$ SI COP=95 COMPARE LOGICAL

مقارنة منطقية مباشرة بين القيمة I2 الموجودة في التعليمة و (S1) .

CLC $D_1(L,B_1),D_2(B_2)$ SS COP=D5 COMPARE LOGICAL

مقارنة بين سلاسل تمتد حتى 256 بايتة بطول مشترك L .

CLM $R_1,M_3,D_2(B_2)$ RS COP=BD COMPARE LOGICAL CHARACTERS UNDER MASK
، (370)

القناع M3 ، المكوّن من أربع بتات يختار في R1 من 0 إلى 4 بايتات تُقارن بالبايتات المتتالية إنطلاقاً من العنوان S2 . البتة الأولى من القناع ، إذا كانت معادلة لـ 1 تختار البايتة الأولى من R1 وهكذا دواليك .

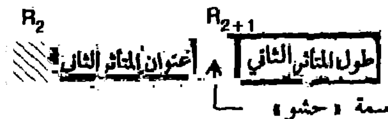
يتم تركيز CC .

القناع المعادل لـ 1011 يختار البايتات 0 ، 2 ، 3 من R1 التي تتم مقارنتها مع ثلاث بايتات إنطلاقاً من S2 . المقارنة تتم من اليسار إلى اليمين .

CLCL R_1,R_2 RR COP=OF COMPARE LOGICAL LONG
(370)

مقارنة بين سلسلتين من البايتات حيث العناوين والأطوال موجودة في المراصف المزدوجة حسب الإتفاق التالي :





تجري العملية من اليسار إلى اليمين من خلال العناوين 1 و2 . إذا لم يكن طول السلسلتين متعادلاً ، يُفترض بأن يُكمَّل الأقصر من اليمين بالسمة «padding» (سمة الحشو) .

العملية تتم بايتة بعد بايتة مع زيادة عناوين وتقصير الطول . وهي قابلة للإنقطاع بين مقارنة بايتين . وتتوقف عند أول لا معادلة نلتقيها أو في نهاية السلسلة مع تركيز كود ـ الشرط .

4.13 . مقارنات منطقية خاصة

لقد قمنا هنا بتصنيف التعليمات التي ، زيادة عن وظيفتها في المقارنة ، تتمتّع بعمل خاص . هذه التعليمات تركّز كود الشرط بصورة مختلفة .

CS $R_1,R_3,D_2(B_2)$ RS COP=BA COMPARE AND SWAP
(370)

مقارنة بين R_1 و (S_2)

إذا : $R_1 = (S_2)$. عندئذٍ $R_3 \rightarrow (S_2)$ و $0 \rightarrow CC$.

إذا : $R_1 \neq (S_2$ عندئذٍ $R_1 \rightarrow (S_2)$ و $1 \rightarrow CC$.

CDS $R_1,R_3,D_2(B_2)$ RS COP=BB COMPARE DOUBLE AND SWAP
(370)

مقارنة بين R_1 و(S_2)

إذا : $R_1 = (S_2)$ عندئذٍ $R_3 \rightarrow (S_2)$ و$0 \rightarrow CC$

إذا : $R_1 \neq (S_2)$ عندئذٍ $R_1 \rightarrow (S_2)$ و $1 \rightarrow CC$.

المقارنة CDS تتم على 64 بتة . وبالنتيجة فإن R_1 وR_3 هما مرصفان مزدوجان (فقرة 1.10) و S_2 هو عنوان كلمة مزدوجة من الذاكرة .

تُستعمل هاتين التعليمتين لتنفيذ المزامنة بين مهمتين تقتسمان منطقة مشتركة من الذاكرة . عندما تتم المعادلة ، فإن كل بلوغ للعنوان S_2 هو ممنوع لأي مُعالج مركزي حتى نهاية عملية النقل $R_3 \rightarrow (S_2)$.

TM $D_1(B_1),I_2$ SI COP=91 TEST UNDER MASK

TM تقوم باختبار حالة البتات من البايتة ذات العنوان S_1. I_2 هي قناع من 8 بتات . كل «1» ، موجود في القناع يسمح باختبار وجود بتة «1» في الموقع المناسب من البايتة S_1 .

مثلاً : القناع X'60' أي B'01100000' يفحص وجود «1» في الموقعين 1 و2 من البايتة . ويجري إهمال المواقع الأخرى . وفي الإجمال ، فإن TM يقوم بتنفيذ عملية AND منطقية بين البايتة التي تم فحصها والقناع دون تعديل البايتة ولكن بتركيز كود الشرط فقط :

CC = 0 : جميع البتات التي جرى إختبارها هي 0 أو القناع هو في صفر ،
CC = 1 : بعض البتات هي صفر ، وأخرى هي 1 ،
CC = 2 : غير مستعمل
CC = 3 : جميع البتات المختبَرة هي 1 .

البايتة المختبرة	11001110	11001110	11001110
القناع	01011100	11001000	00110000
AND	-1-011--	11--1---	--00----
CC	1	3	0

تطبيق :

TM يبدو وكأنه ينتمي إلى CLI . وفعلاً فإن TM يُعتمد لاختبار البتات أكثر من البايتات . مثلاً ، لمعرفة ما إذا كانت البايتة هي رقمية نستعمل CLI لأن القيمة يجب أن تكون محصورة بين F0 وF9 .

TM يمكن أن تُستعمل لتنفيذ تأشير متعدّد .

تمارين :

نذكر أن الدالة «AND» تسمح بجعل البتات تعادل صفراً ، وإن الدالة «OR» تسمح بجعلها 1 وإن «EOR» تسمح بعكسها .

تمرين 1.13 . ضع في صفر ثنائي منطقة بطول L ⩾ 256 بايتة ، مرصفاً ، بايتة .

تمرين 2.13 . اكتب التعليمة التي تسمح بتركيز قيمة كود الطول في تعليمة من نوع SS .

تمرين 3.13 . بدّل مضمون منطقتين من الذاكرة ، مرصفين ، ربعيين من البتات من نفس البايتة .

تمرين 4.13 . تعرّف ما إذا كانت منطقة من الذاكرة مملوءة بفراغ أو بصفر ثنائي

تمرين 5.13 . قم بإجراء تأشير يؤدي إلى تفريع مرّة على اثنتين بواسطة تحويل منطقة قناع تعليمة ...,BC 15 إلى ...,BC 0 .

تمرين 6.13 . قم بإجراء تأشير يؤدي إلى تفريع إلى جميع نقاط العبور ما عدا الأول .

تمرين 7.13 . بدّل جميع أصفار اليسار (X'F0') في عدد عشري بفراغات (X'40') .

تمرين 8.13 . البايتة تسمح بتجميع حتى ثمانية مؤشرات ثنائية . لنأخذ البايتة INDIC التي تجمع المؤشرات الثنائية INDLEC ، INDECR ، IND WAIT المناسبة على التوالي للقيم السادس عشرية X'80' ، X'40' وX'20' من INDIC (تحتل المؤشرات البتات 0 ، 1 و2 من INDIC) . اكتب التعليمات التي تسمح :

- بتعريف INDIC ، INDLEC ، INDECR ، INDWAIT ؛
- بتركيز INDWAIT في 1 ؛
- بتركيز INDLEC وINDWAIT في 1 ؛
- بتركيز INDLEC وINDECR في صفر ؛

- بتفريع إلى ALPHA إذا كانت INDWAIT في «1» ؛
- بتفريع إلى BETA إذا كانت INDWAIT وINDLEC في «1» ؛
- بتفريع إلى GAMMA عندما يكون فقط INDLEC أو INDWAIT في «1» ؛
- بتفريع إلى DELTA عندما تكون INDWAIT وINDLEC في صفر .

لنفترض بأنّنا نرغب بربط INDLEC بالبتة 7 من INDIC بدلاً من البتة 0 ، مما يتناسب مع 'X'01 بدلاً من 'X'80 . الحلّ الخاص بكم هل يسمح بعدم تعديل تعليمات التركيز والاختبار لِـ INDLEC؟

14 . عمليات الازاحة (Shift)

1.14 . التعليمات « المنطقية » والتعليمات « الحسابية »

عند دراسة تعليمات الجمع بفاصلة ثابتة ، لاحظنا ، أنه الى جانب التعليمات A ، AR وAH ، تأتي عمليات الجمع المنطقية . الفرق بين هذين النوعين من العمليات هو التالي :

- تميِّز العمليات الجبرية البتة 0 ، المعتبرة كإشارة ، تجري العملية على 31 بتة مع مُرحِّل محتمل إلى بتة الإشارة . يجري اختيار الإشارة ويمكن أن تؤدي الى إنقطاع من نوع overflow .
- العمليات من نوع منطقي لا تأخذ بعين الإعتبار أي تمييز للبتة ذات الوزن الأكبر . تجري معالجة جميع البتات بنفس الطريقة . أي ترحيل في نهاية البتة ذات الوزن الأكبر لا يؤدي إلى انقطاع .

الإزاحة هي عبارة عن نقل إلى اليسار أو إلى اليمين لعدد n من المواقع لتشكيلة ثنائية موجودة في مرصف بسيط (إزاحة بسيطة) أو في مرصف مزدوج (إزاحة مزدوجة) .

عند الإزاحة تضيع البتات المطرودة . والبتات الداخلة لجهة اليمين هي دائماً صفر . أمّا البتات التي تدخل من اليسار فيمكن أن تكون إما «0» (إزاحة منطقية إلى اليمين أو إزاحة حسابية إلى اليمين لعدد إيجابي) أو «1» (إزاحة جبرية إلى اليمين لعدد سلبي) . سنرى السبب لاحقاً .

2.14 . الإزاحة الجبرية

تجري الإزاحة الجبرية على القيمة ، أي على 31 بتة (إزاحة بسيطة) أو على 63 بتة (إزاحة مزدوجة) .

- الإزاحة إلى اليمين تؤدي إلى إدخال بتات معادلة لبتة الإشارة .

- الإزاحة إلى اليسار تؤدي إلى إدخال 0. إذا جرى تعديل بتة الإشارة سيحدث إنقطاع من نوع overflow بفاصلة ثابتة.

الإزاحة الجبرية تؤدي إلى تركيز كود الشرط على الشكل التالي :

CC = 0	إذا كانت النتيجة صفرا .
CC = 1	إذا كانت النتيجة سلبية .
CC = 2	إذا كانت النتيجة إيجابية
CC = 3	إذا كان يوجد overflow (تعديل في بتة الاشارة في حالة إزاحة إلى اليسار) .

أمثلة :

لتبسيط العرض سنفترض إن حجم المرصف يعادل ثمان بتات . البتة ذات الوزن الأكبر هي إذاً بتة الإشارة .

	مرصف بسيط		مرصف مزدوج	
قبل الإزاحة	00001111 (S)	= +15	00000000 11001111 (S)	= +207
بعد الإزاحة لجهة اليسار ثلاثة	01111000 (S)	= +120 CC = 2	00000110 01111000	= +1656
بعد الإزاحة لجهة اليمين ثلاثة	00000001 (S)	= +1 CC = 2	00000000 00011001 (S)	= +25

قبل الإزاحة	11100101 (S)	= −27
بعد الإزاحة لجهة اليمين 1	11110010 (S)	= −14 CC = 1
بعد الإزاحة بجهة اليسار 4	01010000 (S)	= +80 CC = 3 OVERFLOW

3.14 . الإزاحة المنطقية

تعالج الإزاحة المنطقية 32 بتة (إزاحة بسيطة) أو 64 بتة (إزاحة مزدوجة) دون أخذ بالاعتبار البتة ذات الوزن الأكبر . البتات الداخلة هي دائماً «0» . لا يحدث إنقطاع من نوع overflow . لا يجري تعديل في CC .

أمثلة : على ثمان بتات .

قبل الإزاحة	10011100
بعد الإزاحة لجهة اليسار 2	01110000
بعد الإزاحة لجهة اليمين 2	00100111

4.14 . تعليمات الإزاحة

يوجد أربع عمليات إزاحة جبرية ، أربع تعليمات إزاحة منطقية ، وتعليمة إزاحة لعدد عشري . سنرى هذه الأخيرة عند دراسة الحساب العشري .

الإزاحة الجبرية :

SLA	$R_1,D_2(B_2)$	RS	COP=8B	SHIFT LEFT SINGLE إزاحة بسيطة إلى اليسار
SLDA	$R_1,D_2(B_2)$	RS	COP=8F	SHIFT LEFT DOUBLE إزاحة مزدوجة إلى اليمين
SRA	$R_1,D_2(B_2)$	RS	COP=8A	SHIFT RIGHT SINGLE إزاحة بسيطة إلى اليمين
SRDA	$R_1,D_2(B_2)$	RS	COP=8E	SHIFT RIGHT DOUBLE إزاحة مزدوجة إلى اليمين

الإزاحة المنطقية

SLL	$R_1,D_2(B_2)$	RS	COP=89	SHIFT LEFT SINGLE LOGICAL إزاحة بسيطة منطقية الى اليسار
SLDL	$R_1,D_2(B_2)$	RS	COP=8D	SHIFT LEFT DOUBLE LOGICAL إزاحة منطقية مزدوجة إلى اليسار

SRL $R_1,D_2(B_2)$ RS COP=88 SHIFT RIGHT SINGLE LOGICAL
إزاحة بسيطة منطقية إلى اليمين

SRDL $R_1,D_2(B_2)$ RS COP=8C SHIFT RIGHT DOUBLE LOGICAL
إزاحة مزدوجة منطقية إلى اليمين.

قواعد مشتركة للإزاحات المنطقية والجبرية

- تتم عمليات الإزاحة على مضمون المرصف R_1 .
- بالنسبة لعمليات الإزاحة المزدوجة ، فإن R_1 يجب أن يكون مرصفاً مزدوجاً حسب الإتفاق العادي (فقرة 2.10) .
- المتأثر الثاني $D_2(B_2)$ ليس عنواناً :

1 ـ إذا كان B_2 هو المرصف 0 ، فإن البتات الستّ ذات الوزن الأضعف للنقلة تعطي عدد المواقع المطلوب إزاحتها . SLA 5,3 أو SLA 5,3(0) هما عمليّتا إزاحة لجهة اليسار لثلاثة مواقع ثنائية .

2 ـ إذا لم يكن B_2 هو المرصف 0 ، فإن المرصف المذكور يحتوي على عدد المواقع المطلوب إزاحتها . ونحصل على الإزاحة بشكل غير مباشر. SRDL 6,0(5) يزحل منطقياً المرصف المزدوج (المرصفان 6 و7) لعدد المواقع المشار إليها في المرصف 5 .

- وحدها عمليات الإزاحة الجبرية تقوم بتركيز كود الشرط CC حسب اتفاق الفقرة 2.14 .

تمارين :

تمرين 1.14 ـ ضع في صفر مرصفاً بواسطة الإزاحة .

تمرين 2.14 ـ إضرب واقسم عدداً موجوداً في مرصف على قوة لـ 2 بواسطة الإزاحة .
إفحص ، بالنسبة للقسمة ، إتجاه التقريب .

تمرين 3.14 ـ إفحص فيما إذا كان زوج من المراصف مزدوج / مفرد هو صفر .

تمرين 4.14 ـ برمج إزاحة دائرية لمرصف بسيط.

15 . مسائل

1.15 . الفرز

يتعلّـق ذلك بترتيب جدول من الكلمات التي تحتوي على أعداد بفاصلة ثابتة بترتيب تصاعدي . لقد قمنا باختيار الخوارزم الكلاسيكي الذي يُعرف بـ « طريقة الفقاعة » . تقوم الطريقة على فحص عناصر الجدول من اليسار إلى اليمين مع تبديل العناصر المتتالية الموجودة بشكل عشوائي . نضع إلى اليمين العنصر الأكبر كما نلاحظ من المثل التالي :

```
5 1 3 2
1 5 3 2
1 3 5 2
1 3 2 5
```

إذا كان N هو حجم الجدول ، نبدأ العملية باعتماد الجدول الثانوي بالحجم N −1 وهكذا دواليك ، طالما يوجد عملية تبديل واحدة على الأقلّ خلال التكرار السابق .

ولو إفترضنا أنه خلال فحص الأعداد ، لم تجر أية عملية تبديل فمعنى ذلك إن الترتيب قد حصل .

البرنامج مؤلف من حلقتين BCL1 وBCL2 متداخلتين . الحلقة الداخلية BCL2 تفحص الجدول باستعمال مرصف مؤشر PTR : (PTR) هو عنوان العنصر . العناصر التي جرت مقارنتها هي إذاً ((PTR)) و[1](4+(PTR)) . يتم إنشاء الحلقة بواسطة BXLE . المرصف المزدوج INCRE/REFER يحتوي على الزيادة 4 والحدّ TAB+(N−1) ✱ 4 .

عند إجراء تبديل نقوم بتركيز البايتة INDIC في 1 . الحلقة BCL1 تُكرّر BCL2 طالما إن INDIC=1 .

---

(1) نذكّر بأنه حسب الترميز المعتمد ، (PTR) يُقرأ « مضمون PTR » وهنا هو إذن عبارة عن عنوان . مضمون هذا العنوان ، أي العنصر المطلوب ، يُرمز إليه بـ ((PTR)) .

```
LOC    OBJECT CODE      ADDR1 ADDR2  STMT   SOURCE STATEMENT

000000                                 1 TRI        CSECT
                                       2            PRINT   NOGEN,DATA
                               00002   3 PTR        EQU     2                     POINTEUR SUR ELEMENT DE TAB
                               00004   4 INCRE      EQU     4                     REG INCREMENT POUR BXLE
                               00005   5 REFER      EQU     5                     REGISTRE REFERENCE POUR BXLE
                               00007   6 WORK       EQU     7                     REGISTRE DE TRAVAIL

000000                                 8 SNAPDEB    DS      0H
000000                                 9 PROLOGUE   DS      0H
000000 90EC D00C         0000C        10            STM     14,12,12(13)
                               00000  11            USING   TRI,12                BASE = REGISTRE 12
000004 18CF                           12            LR      12,15
000006 50D0 C0F0         000F0        13            ST      13,SAVE+4
00000A 41D0 C0EC         000EC        14            LA      13,SAVE

                                      16 * INITIALISATION
                                      17            OPEN    (POUB,(OUTPUT))
00001A 4140 0004         00004        23            LA      INCRE,4               LONGUEUR DU MOT
00001E 4150 C0E8         000E8        24            LA      REFER,TAB+(N-1)*4
                                      25            SNAP    DCB=POUB,PDATA=(REGS,PSW,SA),STORAGE=(SNAPDEB,SNAPFIN)
00004E                                46 BCL1       DS      0H
                                      47 *          BOUCLE DE BALAYAGE DU TABLEAU
00004E 9500 C134         00134        48            CLI     INDIC,X'00'           SI PAS DE PERMUTATION ALORS
000052 4780 C082         00082        49            BE      EPILOGUE              TAB EST TRIE

000056 9400 C134         00134        51            NI      INDIC,X'00'           RAZ INDIC
00005A 4120 C0C4         000C4        52            LA      PTR,TAB               INITIALISATION POINTEUR
00005E 1854                           53            SR      REFER,INCRE           INITIALISATION REFERENCE BXLE

000060                                55 BCL2       DS      0H
000060 5872 0000         00000        56            L       WORK,0(PTR)           ELEMT GAUCHE DS REG DE TRAVAIL
000064 5972 0004         00004        57            C       WORK,4(PTR)           COMPARAISON
000068 4700 C07A         0007A        58            BNH     FINBCL2

00006C D203 2000 2004    00000 00004  60            MVC     0(4,PTR),4(PTR)       PERMUTATION
000072 5072 0004         00004        61            ST      WORK,4(PTR)
000076 9601 C134         00134        62            OI      INDIC,X'01'           POSITIONT INDIC

00007A                                64 FINBCL2    DS      0H
00007A 8724 C060         00060        65            BXLE    PTR,INCRE,BCL2

00007E                                67 FINBCL1    DS      0H
00007E 47F0 C04E         0004E        68            B       BCL1
000082                                70 EPILOGUE   DS      0H
                                      71            SNAP    DCB=POUB,PDATA=(REGS,PSW,SA),STORAGE=(SNAPDEB,SNAPFIN)
                                      91            CLOSE   (POUB)
0000BA 58D0 C0F0         000F0        97            L       13,SAVE+4
0000BE 98EC D00C         0000C        98            LM      14,12,12(13)
0000C2 07FE                           99            BR      14
                                     101 * ZONE DES DONNEES
                               0000A 102 N          EQU     10                    NB D'ELEMENTS DE TAB
0000C4 00000000FFFFFFFD              103 TAB        DC      F'15,-3,2,7,11,-1,0,9,2,10'
0000CC 0000000200000007
0000D4 00000008FFFFFFFF
0000DC 0000000000000009
0000E4 000000020000000A
0000EC                               104 SAVE       DS      18F                   SAVE AREA
000134 01                            105 INDIC      DC      X'1'                  INITIALISATION DE INDIC
000136                               106 SNAPFIN    DS      0H
                                     107 POUB       DCB     DSORG=PS,RECFM=VBA,MACRF=(W),BLKSIZE=882,
                                                            LRECL=125,DDNAME=POUBELLE
000000                               155            END     TRI
```

قبل الفرز

```
PSW AT ENTRY TO SNAP   078D1000 0008705E        ILC 2   INTC 0033

REGS AT ENTRY TO SNAP
     FLTR 0-6      0000000000000000     0000000000000000          0000000000000000     0000000000000000

     REGS 0-7      000001A0  80087038  80084F64  00087010         00000004  000870F8  00084F88  000000FF
     REGS 8-15     00000000  00084EB0  00084FD0  00087000         00087010  000870FC  0000C758  00000000

-STORAGE                                             90ECD00C 18CF50D0 C0F041D0 C0EC0700   *           .........O......*
087000                                               C0E80700 4510C048 00720000 0000A400   *........... .....Y.............*
087020    4510C018 8F087148 0A134140 00044150        00087010 80087146 9680101C 0A339500   *................................*
087040    00087148 00000000 00087050 00000000        58720000 59720004 47D0C07A D2032000   *A.......A....D.............K...*
087060    C1344780 C0829400 C1344120 C0C41B54        C04E0700 4510C0A8 00720000 0000A400   *........A......O...............*
087080    20045072 00049601 C1348724 C06047F0        00087010 00087146 9680101C 0A330700   *...............................*
0870A0    00087148 00000000 000870B0 00000000        D00C07FE 0000000F FFFFFFFD 00000002   *............O..................*
0870C0    4510C088 80087148 0A1458D0 C0F098EC        00000009 00000002 0000000A 00000000   *...............................*
0870E0    00000007 0000000B FFFFFFFF 00000000        00000000 00000000 00000000 00000000   *...............................*
087100    00087FB8 00000000 00000000 00000000        00000000 00000000 00000000 00000000   *...............................*
087120    00000000 00000000 00000000 00000000                                              *.........                      *
087140    00000000 01000000
```

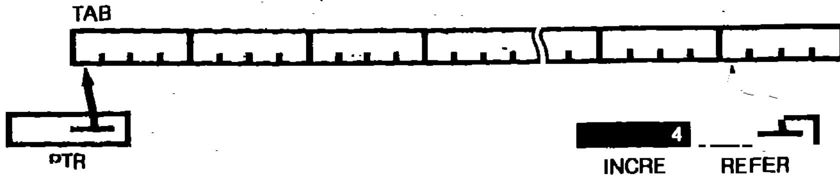
بعد الفرز

```
PSW AT ENTRY TO SNAP   078D1000 000870BE        ILC 2   INTC 0033

REGS AT ENTRY TO SNAP
     FLTR 0-6      0000000000000000     0000000000000000          0000000000000000     0000000000000000

     REGS 0-7      008DC698  80087098  000870E8  00087010         00000004  000870E4  00084F88  00000002
     REGS 8-15     00000000  00084EB0  00084FD0  00087000         00087010  000870FC  0000C758  00000000

-STORAGE                                             90ECD00C 18CF50D0 C0F041D0 C0EC0700   *           .........O......*
087000                                               C0E80700 4510C048 00720000 0000A400   *........... .....Y.............*
087020    4510C018 8F087148 0A134140 00044150        00087010 80087146 9680101C 0A339500   *................................*
087040    00087148 00000000 00087050 00000000        58720000 59720004 47D0C07A D2032000   *A.......A....D.............K...*
087060    C1344780 C0829400 C1344120 C0C41B54        C04E0700 4510C0A8 00720000 0000A400   *........A......O...............*
087080    20045072 00049601 C1348724 C06047F0        00087010 80087146 9680101C 0A330700   *...............................*
0870A0    00087148 00000000 00087050 00000000        D00C07FE FFFFFFFD FFFFFFFF 00000000   *............O..................*
0870C0    4510C0B8 80087148 0A1458D0 C0F098EC        0000000A 0000000B 0000000F 00000000   *...............................*
0870E0    00000002 00000002 00000007 00000009        00000000 00000000 00000000 00000000   *...............................*
087100    00087FB8 00000000 00000000 00000000        00000000 00000000 00000000 00000000   *...............................*
087120    00000000 00000000 00000000 00000000                                              *.........                      *
087140    00000000 00000000
```



من الممكن أن نكتب الخوارزم على الشكل التالي :

('←' هو رمز التخصيص)

```
          INCRE ← 4
          INDIC ← 1
          REFER ← TAB+(N-1)*4
    BCL1: TANT QUE INDIC ≠ 0 FAIRE
             INDIC ← 0
             REFER ← REFER - INCRE
             BCL2:    ------
                      exploration
                      ------
             FIN BCL2
    FIN BCL1
```

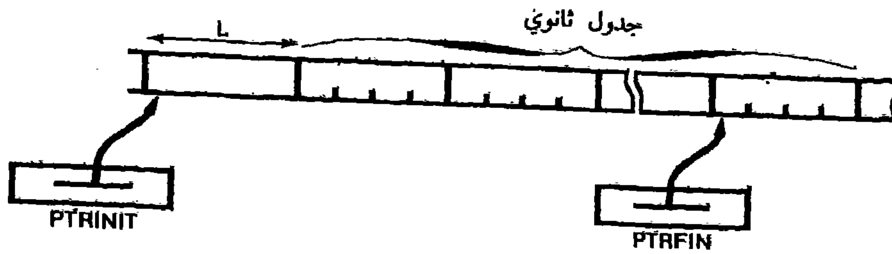
SNAP هي عبارة عن ماكرو تعليمة نموذجية تسمح بالحصول على صورة سادس عشرية من الذاكرة . إستعمالها يتطلب فتح السجل (OPEN) ، إغلاق (CLOSE) ووصف السجل بواسطة الماكرو تعليمة DCB. PRINT NOGEN (سطر 2) تسمح بإلغاء توليد كود الماكرو تعليمات .

2.15 . إستشارة فرقانية للجدول

يقوم البرنامج على البحث عن وجود أو غياب معلومة من داخل أحد الجداول . البحث المتسلسل يبدو صعباً ويستهلك كثيراً من الوقت عندما يصبح حجم الجدول كبيراً . من الممكن أن نستعمل طريقة الفرقان عندما تكون العناصر منظمة . والصيغة هي التالية :

لنفترض جدولاً TAB من N عنصر منظّم نبحث فيه عن موقع المعلومة الموجودة في MOT . نقوم باستشارة العنصر الموجود في وسط TAB ونقارنه بـ MOT . البحث ينتهي عندما نجد التعادل . وإلا نُعيد الكرّة ونتابع الاستشارة باختيار واحد من الجدولين الثانويين المشكّلين بواسطة القسمة السابقة حسب موقع العنصر الذي نبحث عنه بالنسبة للعنصر الوسط . بعد كل إستشارة تضيق الفسحة التي نبحث فيها إلى النصف .

سنفترض إن طول العنصر هو L وهذا الطول يعادل قوة (أس) P للعدد 2 ($L=2^P$) . هذا سيسمح بإجراء عمليات ضرب وقسمة بواسطة الإزاحة . سنستعمل مراصف مؤشرات لبلوغ العناصر. PTRINIT سيحتوي على عنوان العنصر الأول من الجدول الثانوي ناقص PTRFIN.L سيحتوي على عنوان العنصر الأخير من الجدول الثانوي .



عدد العناصر هو إذاً : $\frac{PTRFIN - PTRINIT}{L}$

عنوان العنصر الوسط هو :

عنوان البداية + $\frac{1}{2}$ × عدد العناصر × L

أي : $PTRINIT + L + \frac{1}{2}\left(\frac{PTRFIN - PTRINIT}{L}\right) \times L$

عند القسمة على L يجب إهمال الباقي الذي قد يظهر .

البرنامج التالي جرى اختباره بعد إجراء نداء لبرنامجين ثانويين مكتوبين بلغة فورتران : LIRE وECR . وجود نداءات بلغة فورتران من خلال برنامج رئيسي بلغة المؤول يتطلب كتابة التعليمات 59 و60 غير الموجودة إذن إلا لأسباب توافقية بإشراف النظام المستعمل (FORTRAN G, OS-VS2) .

```
  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
000000                                37 DICHO    START 0
                                      38 *
                             00005    39 PTRINIT  EQU   5                   POINTEUR DEBUT DE SOUS-TABLE
                             00002    40 PTRFIN   EQU   2                   POINTEUR  FIN DE SOUS-TABLE
                             00003    41 PTRELEM  EQU   3                   POINTEUR MILIEU ET RANG
                             00006    42 WORK     EQU   6                   REGISTRE DE TRAVAIL
                             00004    43 L        EQU   4                   LONGUEUR DE L'ELEMENT
                             00002    44 P        EQU   2                   L = 2**P

                                      46          PROLOGUE ABASE=DICHO,RBASE=(12)
                             00000    47+PROLOG   EQU   *
000000 90EC D00C       0000C          48+         STM   14,12,12(13)
                             00000    49+         USING DICHO,12
000004 18CF                           50+         LR    12,15
000006 50D0 C01C       0001C          51+         ST    13,SAVEAREA+4
00000A 182D                           52+         LR    2,13
00000C 4100 C018       00018          53+         LA    13,SAVEAREA
000010 50D2 0008       00008          54+         ST    13,8(2)
000014 47F0 C060       00060          55+         B     *+76
000018                                56+SAVEAREA DS    18F

                                      58          PRINT NOGEN,DATA
000060 58F0 C160       00160          59          L     15,=V(IBCOM#)
000064 45EF 0040       00040          60          BAL   14,64(15)

000068 5880 C164       00164          62          L     8,=F'12'            NB D'ITERATIONS SUR LE PGM
                                      63 LECT     CALL  LIRE,(MOT)

                                      76 * INITIALISATION
000082 4150 C130       00130          77          LA    PTRINIT,TAB-L
000086 4120 C158       0015B          78          LA    PTRFIN,TAB+(N-1)*L

                                      80 * CALCUL ADRESSE ELEMENT MILIEU
00008A                                81 RECHELEM DS    0H
00008A 1832                           82          LR    PTRELEM,PTRFIN
00008C 1835                           83          SR    PTRELEM,PTRINIT
00008E 8830 0003       00003          84          SRL   PTRELEM,P+1         DIVISION PAR 2*L
000092 1233                           85          LTR   PTRELEM,PTRELEM     (PTRELEM)=NB ELEM DS SOUS-TABLE
000094 4770 C09C       0009C          86          BNZ   SUITE
000098 4130 0001       00001          87          LA    PTRELEM,1           SI 0 ON FORCE A 1

00009C 8930 0002       00002          89 SUITE    SLL   PTRELEM,P           MULTIPLICATION PAR L
0000A0 1A35                           90          AR    PTRELEM,PTRINIT
0000A2 D5C3 3000 C12C 00000 0012C     91          CLC   0(L,PTRELEM),MOT    COMPARAISON
0000A8 4720 C0E4       000E4          92          BH    SUP                 BRANCH SI ELEM > (MOT)
0000AC 4740 C0DE       000DE          93          BL    INF                 BRANCH SI ELEM < (MOT)

                                      95 * ON A TROUVE L'ELEMENT. CALCUL DU RANG ELEMENT = (MOT)
0000B0 4160 C130       00130          96          LA    WORK,TAB-L
0000B4 1B36                           97          SR    PTRELEM,WORK
0000B6 8830 0002       00002          98          SRL   PTRELEM,P           DIVISION PAR LONGUEUR
0000BA 5030 C130       00130          99          ST    PTRELEM,RANG
                                     100 * IMPRESSION DU RANG ET DE LA VALEUR
                                     101          CALL  ECR,(MOT,RANG)
0000DA 47F0 C11A       0011A         114          B     EPILOGUE
```

```
  LOC  OBJECT CODE      ADDR1 ADDR2  STMT   SOURCE  STATEMENT
0000DE                                116 INF      DS    0H                                  ELEM < (MOT)
0000DE 1853                           117          LR    PTRINIT,PTRELEM
0000E0 47F0 C0EA        000EA         118          B     TESTFIN

0000E4                                120 SUP      DS    0H                                  ELEM > (MOT)
0000E4 1823                           121          LR    PTRFIN,PTRELEM
0000E6 5B20 C128        00128         122          S     PTRFIN,LONG

0000EA                                124 TESTFIN  DS    0H
0000EA 1552                           125          CLR   PTRINIT,PTRFIN
0000EC 4780 C0F4        000F4         126          BE    NONTROUV
0000F0 47F0 C08A        0008A         127          B     RECHELEM

0000F4                                129 NONTROUV DS    0H                                  PAS TROUVE
0000F4 D703 C130 C130   00130 00130   130          XC    RANG,RANG
                                      131          CALL  ECR,(MOT,RANG)
000116 47F0 C11A        0011A         144          B     EPILOGUE

00011A                                147 EPILOGUE DS    0H
00011A 4680 C06C        0006C         148          BCT   8,LECT
00011E 58D0 C01C        0001C         149          L     13,SAVEAREA+4
000122 98EC D00C        0000C         150          LM    14,12,12(13)
000126 07FE                           151          BR    14

                                      153 * ZONE DE DONNES
                              0000A   154 N        EQU   10                                  NB DE MOTS DE LA TABLE
000128 00000004                       155 LONG     DC    F'4'                                LONGUEUR D'UN ELEMENT
00012C                                156 MOT      DS    F
000130                                157 RANG     DS    F
000134 0000000100000002               158 TAB      DC    F'1,2,4,8,16,32,64,128,256,512'
00013C 0000000400000008
000144 0000001000000020
00014C 0000004000000080
000154 0000010000000200
000000                                159          END   DICHO
00016C 00000000                       160                =V(IBCOM#)
000164 0000000C                       161                =F'12'
```

16 . الحساب العشري

1.16 . عموميات

تقدِّم التعليمات الحسابية العشرية وسائل لإجراء الحسابات على الأعداد العشرية « المتراصة packed » التي رأيناها في الفقرة 3.5.2.ج. ولاحقاً سندرس عملية تحويلها لمعطيات .

التعليمات الحسابية هي بنسق SS وتستعمل الطولين L_1 وL_2 للمتأثرين . يبقى طول المتأثرات محدوداً بـ 16 بايتة (31 رقماً عشرياً زائد الإشارة في التمثيل المتراص و16 رقماً وإشارة في التمثيل الموسَّع) لأنها تقسِّم المنطقة L بالنسق SS . شكل هذه التعليمات هو التالي :

COP	L_1	L_2	B_1	D_1	B_2	D_2

ونشير إلى أنّـه جرت العادة بالنسبة للتعليمات SS بأن تكون القيم المؤوّلـة في المناطق L هي بالطول المذكور في تعليمة مؤول ناقص 1 . هكذا ، فالتعليمة :

A P ALPHA (16), BETA (10)

سيتم تأويلها مع القيم الثنائية 1111 و1001 بالنسبة للطول .

تضع التعليمات الحسابية النتيجة في المتأثر الأول الذي يتم إلغاؤه ويجب أن يكون هذا المتأثر بطول كافٍ لاستيعاب النتيجة دون حدوث overflow وقطع للعدد . يظهر overflow إذا لم يكن المتأثر الأول بالطول المناسب لاستيعاب النتيجة . عندما تكون $L_1<L_2$ لا يحدث overflow إذا لم يكن هناك مُرحِّـل (carry) خارج الإمكانيات المقدّمة من الطول L_1 . ويمكن تقنيع overflow بواسطة البتة SPM .

عند إجراء العمليات ، فإن الفاصلة لا تُمثّـل والتراصف يتم لجهة اليمين ، كما يمكن حصر المتأثـرات بواسطة عمليات إزاحة عشرية مناسبة .

تتحقّق الدارات ، خلال التنفيذ ، من صلاحية الأرقام العشرية والإشارات . والتقاء عنصر غير صالح يؤدي إلى انقطاع من نوع استثناء بالمعطيات .

المتأثـرات 1 و2 يمكن أن تندمج بشرط أن تكون بنفس المواقع (متراصفة) بالنسبة للبايتات ذات الوزن الأضعف . من الممكن هكذا إضافة عدد إلى نفسه :

مثلاً :

0 0	0 1	2 3	4 5	6 S

بعنوان ALPHA

التعليمة :

AP ALPHA(5),ALPHA+3(2) تجمع S123456 إلى S456

يتم تركيز كود الشروط CC حسب إشارة النتيجة .

2.16 . التعليمات

AP $D_1(L_1,B_1),D_2(L_2,B_2)$ SS COP=FA ADD DECIMAL
$(S_1) + (S_2) \rightarrow (S_1)$
يتم تركيز كود الشرط CC .

ZAP $D_1(L_1,B_1),D_2(L_2,B_2)$ SS COP=F8 ZERO AND ADD
$(S_2) \rightarrow (S_1)$
تعادل العملية جمع عدد إلى صفر . ويتم تركيز CC .

SP $D_1(L_1,B_1),D_2(L_2,B_2)$ SS COP=FB SUBTRACT DECIMAL
$(S_1) - (S_2) \rightarrow (S_1)$
تركيز CC .

MP $D_1(L_1,B_1),D_2(L_2,B_2)$ SS COP=FC MULTIPLY DECIMAL
$(S_1) \times (S_2) \rightarrow (S_1)$
يجب أن نحصل على : $L_2 \leqslant 8$ و $L_2 < L_1$ وإلا سيحدث انقطاع .
CC يبقى بدون تعديل .

DP $D_1(L_1,B_1),D_2(L_2,B_2)$ SS COP=FD DIVIDE DECIMAL
$(S_1) : (S_2) \rightarrow (S_1)$
يجري وضع النتيجة إلى اليسار في (S_1) . الباقي يُخزَّن إلى اليمين في (S_1) وبنفس طول S_2 .

حجم نتيجة القسمة هو 8 بتات : $L_2 - L_1$ يجب أن نحصل على $L_2 \leqslant 8$ و $L_2 < L_1$ وإلا سيحدث انقطاع[1] CC بدون تعديل .

---

(1) إنتباه : يتعلّق ذلك بالطول L بلغة المؤول وليس بطول القيم .

CP $D_1(L_1,B_1),D_2(L_2,B_2)$ SS COP=F9 COMPARE DECIMAL

تجري مقارنة المتأثـرين ويتمّ تعديل مضمون CC . إذا كانت أطوال المتأثرات غير متعادلة ، فإن المنطقة الأصغر يجري ملؤها بصفر لجهة اليسار .

SRP $D_1(L_1,B_1),D_2(B_2),I_3$ SS COP=F0 SHIFT AND ROUND DECIMAL
(370)

يجب الإنتباه إلى النسق الخاص بهذه التعليمة . عند التأويل ، فإن I_3 تأخذ الموقع الطبيعي المحفوظ لـ L_2 .

- S_1 هو عنوان المتأثـر المطلوب إزاحته .
- L_1 هو الطول .
- $D_2(B_2)$ ليس عنواناً : البتات الست ذات الوزن الأضعف والمُعتبرة كعدد صحيح بإشارة ، تدل على اتجاه وعدد الأرقام العشرية المطلوب إزاحتها . ويجري إهمال البتات الأخرى . القيمة السلبية (مكمّـل إلى 2) هي إزاحة إلى اليمين والنتيجة السلبية هي إزاحة إلى اليسار .
- I_3 هو « عامل التدوير » يُستعمل للإزاحات إلى اليمين . تضاف قيمته إلى الرقم المستخرج بالإزاحة إلى اليمين والمُرحَّل المحتمل يرتدّ الى اليسار .
- توضع النتيجة في (S_1) .
- لا تشترك الإشارة بعملية الإزاحة .

---

17 الحساب بفاصلة متحركة

لم يبدُ لنا أساسياً شرح هذه التعليمات بكثير من العناية كما جرى بالنسبة للتعليمات السابقة . فدراسة هذه المجموعة من التعليمات لن تحمل لنا سوى قليلاً من المعلومات الجديدة حول الأوالية الأساسية لتشغيل المكنات ، بينما نحن نهتم بالدرجة الأولى بهذه الأوالية . ولكن المستعمل الذي فهم جيداً كل ما هو سابق لن ينزعج كثيراً من متابعة هذا الفصل . نفترض هنا بأن القارىء قد استوعب قراءة الفقرة 3.5.2 .ب حول الفاصلة المتحركة في تمثيل المعطيات . ولكي نتذكّر بسهولة الكود الحرفي لهذه العمليات ، من الجيّد أن نراجع الفقرة 2.10 المتعلقة بالترميز : الحرف النهائي «R» يختص بالتعليمة RR ، والأحرف E ، U ، D ، W وX هي نسق القصير المعايَر (normalized) ، والقصير غير المعايَر والطويل المعاير والطويل غير المعايَر والموسّع .

1.17 . عموميات

هذه التعليمات تعمل مع المراصف المتحرّكة المرقّمة 0 ، 2 ، 4 و6 بطول 64 بتة . الأعداد بفاصلة متحركة القصيرة توضع في الـ 32 بتة ذات الوزن الأكبر من المراصف خلال العمليات . في هذه الحالة فإن الأوزان الضعيفة يجري إهمالها . الأعداد الطويلة بالفاصلة المتحركة تشغل كامل المراصف والأعداد الموسّعة بفاصلة متحركة تشغل مرصفين متتاليين . يجري تركيز موقع كود الشرط كالعادة :

جدول 1.17

CC	بالنسبة للتعليمات الجبرية	بالنسبة للمقارنات
0	نتيجة صفر	متأثر 1 = متأثر 2
1	نتيجة سلبية	متأثر 1 < متأثر 2
2	نتيجة إيجابية	متأثر 1 > متأثر 2
3		

2.17 التعليمات

نجد نفس الخصائص التي رأيناها لدى معالجة الأعداد بفاصلة ثابتة . في حالة الشك بالإمكان مراجعتها

LER	R_1,R_2	RR	COP=38	LOAD	متأثرات قصيرة
LE	$R_1,D_2(X_2,B_2)$	RX	COP=78	LOAD	متأثرات قصيرة
LDR	R_1,R_2	RR	COP=28	LOAD	متأثرات طويلة
LD	$R_1,D_2(X_2,B_2)$	RX	COP=68	LOAD	متأثرات طويلة
				CC دون تعديل	
LTER	R_1,R_2	RR	COP=32	LOAD AND TEST	متأثرات قصيرة
LTDR	R_1,R_2	RR	COP=22	LOAD AND TEST	متأثرات طويلة
LCER	R_1,R_2	RR	COP=33	LOAD COMPLEMENT	متأثرات قصيرة شحن مع تغير الاشارة
LCDR	R_1,R_2	RR	COP=23	LOAD COMPLEMENT	متأثرات طويلة شحن مع تغير الاشارة
LNER	R_1,R_2	RR	COP=31	LOAD NEGATIVE	متأثرات قصيرة
LNDR	R_1,R_2	RR	COP=21	LOAD NEGATIVE	متأثرات طويلة
LPER	R_1,R_2	RR	COP=30	LOAD POSITIVE	متأثرات قصيرة
LPDR	R_1,R_2	RR	COP=20	LOAD POSITIVE	متأثرات طويلة
				تركيز أو تعديل CC	
LRER (370)	R_1,R_2	RR	COP=35	LOAD ROUNDED	المتأثر 2 الطويل يجري تدويره ووضعه في المتأثر الأول القصير
LRDR (370)	R_1,R_2	RR	COP=25	LOAD ROUNDED	المتأثر الموسع يجري تدويره ووضعه في المتأثر الأول الطويل
				دون تعديل	
STE	$R_1,D_2(X_2,B_2)$	RX	COP=70	STORE	متأثرات قصيرة
STD	$R_1,D_2(X_2,B_2)$	RX	COP=60	STORE	متأثرات طويلة
				CCدون تعديل	
CER	R_1,R_2	RR	COP=39	COMPARE	متأثرات قصيرة
CE	$R_1,D_2(X_2,B_2)$	RX	COP=79	COMPARE	متأثرات قصيرة
CDR	R_1,R_2	RR	COP=29	COMPARE	متأثرات طويلة
CD	$R_1,D_2(X_2,B_2)$	RX	COP=69	COMPARE	متأثرات طويلة
				تركيز أو تعديل CC	
AER	R_1,R_2	RR	COP=3A	ADD NORMALIZED	متأثرات قصيرة
AE	$R_1,D_2(X_2,B_2)$	RX	COP=7A	ADD NORMALIZED	متأثرات قصيرة
ADR	R_1,R_2	RR	COP=2A	ADD NORMALIZED	متأثرات طويلة
AD	$R_1,D_2(X_2,B_2)$	RX	COP=6A	ADD NORMALIZED	متأثرات طويلة
AXR (370)	R_1,R_2	RR	COP=36	ADD NORMALIZED	متأثرات موسعة
				تركيز أو تعديل CC	

AUR	R_1,R_2	RR	COP=3E	ADD UNNORMALIZED(ope	متأثرات قصيرة
AU	$R_1,D_2(X_2,B_2)$	RX	COP=7E	ADD UNNORMALIZED(o	متأثرات قصيرة
AWR	R_1,R_2	RR	COP=2E	ADD UNNORMALIZED (of	متأثرات طويلة
AW	$R_1,D_2(X_2,B_2)$	RX	COP=6E	ADD UNNORMALIZED (op	متأثرات طويلة

تركيز أو تعديل CC

SER	R_1,R_2	RR	COP=3B	SUBTRACT NORMALIZED	متأثرات قصيرة
SE	$R_1,D_2(X_2,B_2)$	RX	COP=7B	SUBTRACT NORMALIZED	متأثرات قصيرة
SDR	R_1,R_2	RR	COP=2B	SUBTRACT NORMALIZED	متأثرات طويلة
SD	$R_1,D_2(X_2,B_2)$	RX	COP=6B	SUBTRACT NORMALIZED	متأثرات طويلة
SXR (370)	R_1,R_2	RR	COP=37	SUBTRACT NORMALIZED	متأثرات موسعة

تركيز أو تعديل CC

SUR	R_1,R_2	RR	COP=3F	SUBTRACT UNNORMALIZED	متأثرات قصيرة
SU	$R_1,D_2(X_2,B_2)$	RX	COP=7F	SUBTRACT UNNORMALIZED	متأثرات قصيرة
SWR	R_1,R_2	RR	COP=2F	SUBTRACT UNNORMALIZED	متأثرات طويلة
SW	$R_1,D_2(X_2,B_2)$	RX	COP=6F	SUBTRACT UNNORMALIZED	متأثرات طويلة

تركيز أو تعديل CC

MER	R_1,R_2	RR	COP=3C	MULTIPLY	متأثرات قصيرة ونتيجة موسعة
ME	$R_1,D_2(X_2,B_2)$	RX	COP=7C	MULTIPLY	متأثرات قصيرة ونتيجة موسعة
MDR	R_1,R_2	RR	COP=2C	MULTIPLY	متأثرات طويلة
MD	$R_1,D_2(X_2,B_2)$	RX	COP=6C	MULTIPLY	متأثرات طويلة
MXDR (370)	R_1,R_2	RR	COP=27	MULTIPLY	متأثرات طويلة ونتيجة موسعة
MXD (370)	$R_1,D_2(X_2,B_2)$	RX	COP=67	MULTIPLY	متأثرات طويلة ونتيجة موسّعة
MXR (370)	R_1,R_2	RR	COP=26	MULTIPLY	متأثرات موسّعة

دون تعديل

DER	R_1,R_2	RR	COP=3D	DIVIDE	متأثرات قصيرة
DE	$R_1,D_2(X_2,B_2)$	RX	COP=7D	DIVIDE	متأثرات قصيرة
DDR	R_1,R_2	RR	COP=2D	DIVIDE	متأثرات طويلة
DD	$R_1,D_2(X_2,B_2)$	RX	COP=6D	DIVIDE	متأثرات طويلة

لا تتغير CC

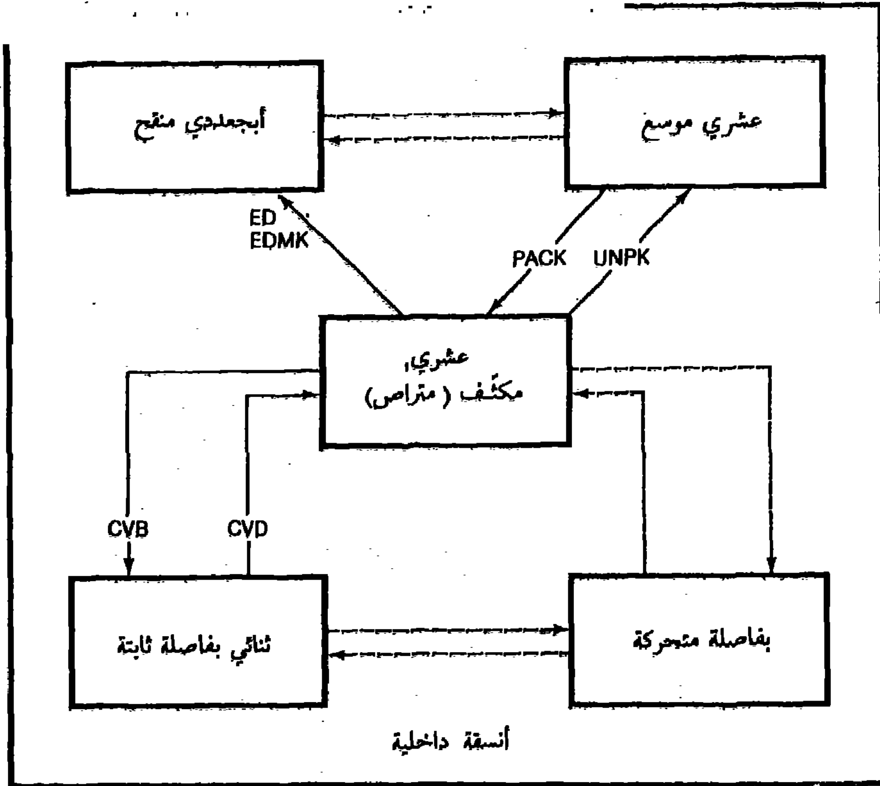
HER	R_1,R_2	RR	COP=34	HALVE	متأثرات قصيرة
HDR	R_1,R_2	RR	COP=24	HALVE	متأثرات طويلة

يُقسم المتأثـر الثاني على 2 وتوضع نتيجة القسمة المعايرة في المتأثر الأوّل .

18 . تعليمات التحويل والتمثيل

1.18 . عموميات

لقد رأينا أن النظام 370 كان يتمتع بثلاث طبقات من الدارات الحسابية العاملة بثلاث طرق مختلفة لتمثيل المعطيات الرقمية . ولكن ، المعطيات الداخلة إلى الذاكرة تكون عادةً مكوَّدة بتمثيل أبجعددي . من هنا ، فإن كل عملية حسابية على معطى رقمي داخل إلى المكنة ، من خلال ناقل بطاقات مثلاً ، يمكن أن تتطلّب عدة عمليات تحويل للتمثيل قبل معالجته بالحساب العشري ، الثنائي أو بفاصلة متحركة . المخطط 1.18 يعرض مختلف الأشكال الداخلية وعمليات النقل الممكنة التي تتم بواسطة هذه التعليمات . الخطوط المنقّطة تمثّل التحويلات التي تجريها برامج متخصّصة .



مخطط 1.18

2.18 . تعليمات التحويل

PACK $D_1(L_1,B_1),D_2(L_2,B_2)$ SS COP=F2 PACK
$(S_2) \rightarrow (S_1)$

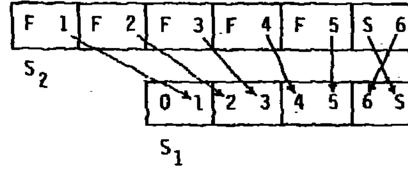
عشري مكثف عشري موسع
(متراص)

هذه التعليمة تحوّل منطقة S2 ، يُفترض إنها عشرية موسّعة ، إلى عشرية متراصة . التحويل يتم من اليمين إلى اليسار بدون تحقق من صلاحية الأكواد .

إذا كانت المنطقة S1 أكبر من الضروري ، فهي تُكمَّل بأصفار (00) لجهة اليسار .

إذا كانت S1 قصيرة جداً يحدث قطع لجهة اليسار .

S1 وS2 يمكن أن تتراكبا .



UNPK $D_1(L_1,B_1),D_2(L_2,B_2)$ SS COP=F3 UNPACK
$(S_2) \rightarrow (S_1)$

عشري موسع عشري مكثّف

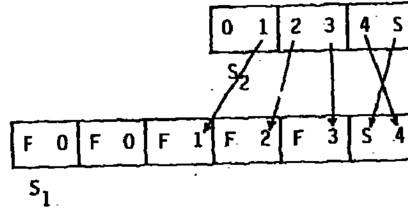
التعليمة تحوّل منطقة S2 ، يفترض إنها عشرية متراصة ، في S1 عشري موسع

التحويل يتم من اليمين إلى اليسار ، بدون تحقق من صلاحية الأكواد .

إذا كانت المنطقة S1 أصغر ، يحدث قطع أو بتر لجهة اليسار .

إذا كانت طويلة تُستكمل بأصفار (F0) لجهة اليسار .

S1 وS2 يمكن أن تتراكبا



CVB $R_1,D_2(X_2,B_2)$ RX COP=4F CONVERT TO BINARY
$(S_2) \rightarrow R_1$

ثنائي عشري متراص

محصورة في كلمة مزدوجة

صلاحية الاشارة والبتات الرقمية في S2 يتم التحقق منها . كل خطأ يؤدي إلى انقطاع .
يفترض بأن تكون S2 عبارة عن عنوان لكلمة مزدوجة بطول 8 بايتات .
يُحدِّد التحويل بالأعداد القصوى والصُغرى التي من الممكن تمثيلها في 32 بتة ، أي :

–2 147 483 648 و +2 147 483 647.

CVD $R_1,D_2(X_2,B_2)$ RX COP=4E CONVERT TO DECIMAL

$R_1 \rightarrow (S_2)$

ثنائي عشري متراص موجود في كلمة مزدوجة

يتألف العدد العشري الحاصل من 15 رقماً إضافةً إلى الإشارة : «C» للجمع (+) و«D» للناقص (–) . يبقى كود الشرط بدون تغيير .

3.18 . التنقيح والطباعة

إنّ مضمون كلمة آلية ثنائية ، لُمعطى عشري أو بفاصلة متحركة يجبْ ، قبل طباعته أن يخضع لتحويل معيَّـن . يجب أن يتم تحويل قيمته الثنائية إلى أكواد من السمات القابلة للطباعة . قد يكون من الضروري إدخال فاصلة ، نقطة عشرية ، إشارة أو سمات تعبئة (حالة طباعة الشيكات) .

يوجد تعليمتان ED وEDMK تحقّقان هذا العمل بتحويل منطقة أولية (عشري متراص) إلى منطقة تنقيح وطباعة .

مثلاً :

منطقة أولية

0 0	1 2	3 4	5 D

منطقة تنقيح

5 C	5 C	5 C	6 0	F 1	F 2	F 3	4 B	F 4	F 5
*	*	*	-	1	2	3	.	4	5

سمات تعبئة إدخال فاصلة عشرية

لكي يتم هذا ، فإن المبرمج يضع في حيِّـز الطباعة قناعاً مؤلّـفاً من :

- سمة تعبئة .
- أكواد تدل على: مواقع الأرقام ، المكان الذي من خلاله يتم تحويل الأصفار «0» بدون ذات معنى ، السمات المطلوب إدخالها في نهاية حقل الطباعة .

هذه التعليمات تعمل بعلاقة مع مؤشر ثنائي يُدعى « مؤشر معني » . يُوضع هذا المؤشر في «1» عندما نلتقي برقم ذي معنى في المنطقة الأولية أو عندما نلتقي مكان الأصفار التي من الواجب تحويلها .

نتعرّف هنا على العمل الجاري بواسطة « صور » الطباعة بلغة كوبول . لن يتم شرح هذه التعليمات هنا وننصح بمراجعة وثائق IBM370.

ED D1(L,B1),D2(B2) SS COP=DE EDIT

S1 : منطقة الطباعة ، بطول L وتحتوي على القناع .
S2 : عنوان المنطقة الأولية (المنبع هو منطقة عشرية متراصة) . يتم تعديل CC حسب إشارة اخر حقل .

EDMK D1(L,B1),D2(B2) SS COP=DF EDIT AND MARK

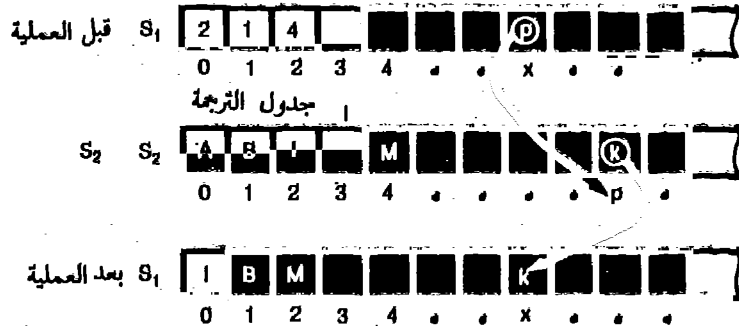
تتمتّع S1 وS2 بنفس المعنى . عنوان الرقم الأول ذي المعنى يُخزّن في المرصف 1 .
يتم تعديل مضمون CC حسب إشارة آخر حقل .

4.18 . الترجمة

TR D1(L,B1),D2(B2) SS COP=DC TRANSLATE

ترجمة سلسلة (S1) بطول L حسب جدول موجود في S2 بطول أقصى يبلغ 256 بايتة .

قبل العملية ، فإن البايتة $S_1+\chi$ $(0 \leq \chi < L)$ S1 تحتوي على الرقم ρ $(0 \leq \rho \leq 255)$ الذي يستخدم كنقطة إدخال إلى الجدول .
بعد العملية : $(S_1+\chi) \leftarrow (S_2+\rho)$ يبقى CC بدون تعديل .



S1 : منطقة البحث بطول L .
S2 : عنوان جدول الترجمة .
التعليمة تستعمل المرصفين 1 و2 .

تُؤخذ البايتة الأولى من المنطقة S1 بعين الاعتبار . كما في TR ، فإنّ قيمته الثنائية تُشكّل نقطة دخول في S2 .

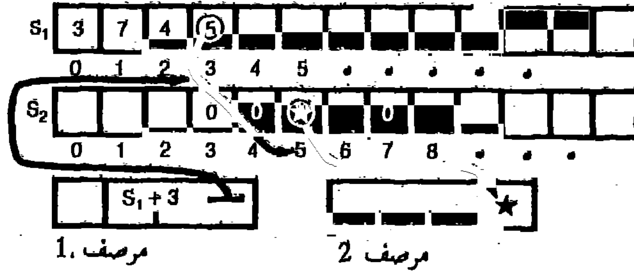
إذا كانت البايتة المناسبة S_2 مختلفة عن صفر فإن قيمتها تُخزَّن في المرصف 2 وعنوان المنطقة التي تسمح بإيجاد التناسب يُخزَّن في المرصف 1 .

وإلّا فإن العملية تتابع مع البايتة التالية من S_1 . يتم تركيز CC :

CC = 0 إذا كانت المنطقة S_1 قد جرى إستكشافها كلياً وجميع البايتات التي جرى إختيارها من S_2 كانت صفراً .

CC = 1 إذا جرى إستكشاف S_1 بشكل جزئي ولم تكن البايتة الأخيرة المختارة صفراً .

CC = 2 إذا جرى إستكشاف المنطقة كلياً وكانت البايتة الأخيرة المختارة مختلفة عن صفر .



CC = 1

في هذا المثل ، لنستطيع متابعة إستكشاف المنطقة ، يجب إعتماد تعديل لعنوان الانطلاق والطول المستكشف .

$R_{1(0-7)}$ و $R_{2(0-23)}$ يبقيان دون تعديل .

S_1 لا يتم تعديلها .

تمارين :

تمرين 1.18 ـ إعادة تنظيم منطقة من الذاكرة .

لنفترض منطقة ARTICLE من 10 بايتات نرغب بنقل البايتات 5 ، 6 ، 7 ، 1 ، 2 إلى المنطقة CLE

ARTICLE	A	B	C	D	E	F	G	H	I	J
CLE	F	G	H	B	C					

إكتب التعليمة التي تسمح بإجراء هذا العمل . في نفس الفكرة نرغب بعكس سلسلة من السمات . هذا النظام يستعمل لاعادة تنظيم مفاتيح الفرز .

تمرين 2.18 ـ لنفترض منطقة مؤلفة من 8 بايتات بقيم ثنائية موجودة بين 0 و15 . نرغب باستبدالها بالكود EBCDIC المناسب للقيم السادس عشرية : سيجري إستبدال 0 بواسطة C'0' ، و10 بواسطة C'A' . . . اكتب التعليمة المناسبة .

هذه الأوالية يمكن أن تستعمل ، بعد عملية تحويل بسيطة ، لطباعة مضمون سادس عشري لكلمة من الذاكرة ، للتحضير للطباعة بواسطة DUMP (دلق) .

19 الانقطاع والادخال والاخراج

(Interruptions and I/0)

1.19 . الانقطاعات

لن يكون موضوعنا تفصيل نظام إدارة الانقطاعات هنا ، ولكن فقط إعطاء القارىء إشارات بالنسبة لطبيعة هذه المسألة . لتفصيلات أكثر تنصح بمراجعة وثائق المُنشيء Principles of operation .

1.1.19 . صيغة الانقطاعات

الانقطاع هو عبارة عن إشارة كهربائية ، مُرسلة من أحد أعضاء النّظام ومعروفة من قِبل الوحدة المركزية . ينتج الانقطاع عن حادثة تتطلب عادةً معالجة مباشرة . لبعض الحوادث صفة خاصة مستعجلة تتطلب تعليق دوران تنفيذ أحد البرامج الجارية كي يتم معالجة الإشارة المُرسلة . في النظام IBM 370 ، الحوادث القادرة على تفريع ووقف تنفيذ البرنامج قد جرى تصنيفها حسب أولوية متناقصة :

- نداء للمشرف (call supervisor) ،
- برنامج ،
- عطل في المكنة ،
- إشارة خارجية ،
- عملية إدخال - إخراج (I/0) ،
- إشارة مؤثّر (operator signal) .

يرتبط بكل فئة درجة إستعجال معيّنة . نتكلّم هنا عن ستة مستويات من الانقطاعات ونظام معالجة الحوادث يجري حسب الأولوية المعتمدة .

2.1.19 . أوالية الإنقطاع

نذكّر بأن المفهوم الذي يدور حوله البرنامج مؤلف من كلمة حالة البرنامج PSW ومن مضمون المراصف العامة والمتحركة المرتبطة به . نشير أيضاً إلى أنه في كل لحظة ، PSW تحتوي على القيمة الحالية لعداد البرنامج . يؤدي تعليق دوران البرنامج أوتوماتيكياً

إلى تخزين مضمون هذه المراصف كي نستطيع معاودة تنفيذ هذا البرنامج المقطوع عند الحاجة . هكذا فالانقطاع يؤدي إلى إطلاق العملية التالية :

1 ـ بشكل أوتوماتيكي (أي بواسطة العتاد (hardware)) ، فإن وصول إشارة الانقطاع تؤدي إلى نسخ PSW الخاصة بالبرنامج الجاري في منطقة محدَّدة من الذاكرة ، تُميِّز فئة الانقطاع . تدعى هذه الكلمة PSW « الكلمة PSW القديمة » .

2 ـ بشكل أوتوماتيكي ، يأخذ العتاد على عاتقه الكلمة الجديدة PSW الموجودة على عنوان من الذاكرة حسب فئة الانقطاع . منذ هذه اللحظة ، يمكن تنفيذ برنامج جديد : وتبدأ معالجة الانقطاع .

3 ـ بعد الإنتهاء من معالجة الانقطاع ، يمكن معاودة العمل بالبرنامج المقطوع وذلك بواسطة إعادة ترميم الكلمة PSW وإعادة تخزين المراصف بالمعلومات التي كان يحتويها قبل قطع البرنامج .

نضيف أن معالجة الانقطاع يمكن أن تُقطع بدورها بواسطة حادثة أكبر أولوية . مجموعة البرامج التي تعالج الانقطاعات تُعتبر جزءاً من نظام التشغيل وتدعى نظام إدارة الانقطاعات .

3.1.19 . قناع الانقطاعات

هذه الأوالية الأساسية يُمكن ، ضمن بعض الشروط ، أن يتم « تقنيعها » بواسطة المبرمج . بواسطة تصفير الأقنعة في الكلمة PSW يُمكن للمبرمج أن يمنع أخذ الحوادث الطارئة بالحسبان . هكذا يمكن إهمال الفيض overflow الناتج عن الحساب وذلك بتركيز القناع المناسب بواسطة التعليمة SPM . الإنقطاع المبرمج المُقنَّع لا يتم أبداً ، كما يوضع الانقطاع المُقنَّع الناتج عن النظام في الانتظار حتى يجري رفع القناع أو القيد عنه . التعليمة SSM التي تسمح بتعديل قناع النظام هي تعليمة خاصّة .

4.1.19 . الانقطاعات الناتجة عن البرنامج

سنعطي هنا أسباب الانقطاعات الناتجة عن البرنامج . وهي تولَّد عادة بسبب خطأ في البرمجة . وتجري الاشارة إليها بواسطة ظهور كود للعودة 0Cx يُدعى «completion code» أو كود الانتهاء .

لتفاصيل أكثر يجب على القارىء أن يراجع وثائق IBM الخاصة .

OPERATION EXCEPTION code = 0C1

ينتج هذا الانقطاع عندما يكون هناك محاولة لتنفيذ تعليمة بكود عملية غير صالح .

PRIVILEGED-OPERATION EXCEPTION code = 0C2

محاولة لتنفيذ تعليمة خاصّة بينما تكون المكنة في صيغة المسألة .

EXECUTE EXCEPTION code = 0C3

التعليمة EX تعود إلى تعليمة أخرى EX .

PROTECTION EXCEPTION code = 0C4

يتعلّق ذلك ببلوغ موقع محمي من الذاكرة .

ADRESSING EXCEPTION code = 0C5

يتعلّق ذلك بمحاولة بلوغ موقع غير موجود في الذاكرة .

SPECIFICATION EXCEPTION code = 0C6

هذا الانقطاع يغطّي أكثر الحالات ، لن نذكر سوى الأكثر شيوعاً . يتعلّق ذلك بمسألة الحدود : لا تحصر التعليمة بحدود نصف كلمة أو معطى غير مسطّر كما تحتاج التعليمة التي تُرجع إليها .

DATA EXCEPTION code = 0C7

يتعلّق ذلك بمشكلة ناتجة عن تعليمة CVB أو تعليمة عشرية .

FIXED-POINT-OVERFLOW EXCEPTION code = 0C8

overflow في تمثيل بفاصلة ثابتة .

FIXED POINT DIVIDE EXCEPTION code = 0C9

يتعلّق ذلك بالقسمة على صفر ، أو بنتيجة قسمة يزيد حجمها عن حجم المرصف أو بتحويل إلى ثنائي (CVB) حيث النتيجه تزيد عن 31 بتة .

DECIMAL-OVERFLOW EXCEPTION code = 0CA

نلتقي هذه التعليمة في عملية على أعداد عشرية ، عندما يتم فقدان البتات ذات الأوزان العليا لأن المنطقة النهائية هي أصغر من أن تحتوي على النتيجة .

DECIMAL-DIVIDE EXCEPTION code = 0CB

يتعلّق ذلك بالقسمة على صفر في عملية بالنظام العشري .

EXPONENT-OVERFLOW EXCEPTION code = 0CC

الأس الخاص بالنتيجة يزيد عن 127 والقسم العشري (mantisse) ليس صفراً .

EXPONENT-UNDERFLOW EXCEPTION code = 0CD

الأس هو سلبي والقسم العشري ليس صفراً .

SIGIFICANCE EXCEPTION code = 0CE

في عملية جمع أو طرح على أعداد بفاصلة متحركة والقسم العشري هو صفر .

FLOATING POINT-DIVIDE EXCEPTION code = 0CF

قسمة على صفر لأعداد بفاصلة متحركة .

5.1.19 . تعليمات مرتبطة بالانقطاعات

SPM R_1 RR COP=04 SET PROGRAM MASK أقنعة البرنامج

$R_{1(2-7)} \rightarrow CC,$



البتات من 2 إلى 7 من المرصف العام R_1 تُخزَّن (البتات 2 و3) في CC وفي (البتتان 4 و7) قناع البرنامج . نشير هنا إلى أن التعليمات BAL وBALR تشحن المرصف $R_{1(2-7)}$ بالكود CC وبقناع البرنامج .

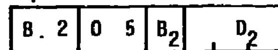
SVC RR COP=0A SUPERVISOR CALL

هذه التعليمة تؤدي إلى انقطاع بكود I . الكلمة القديمة PSW تُخزَّن في الذاكرة على العنوان 32 والكلمة الجديدة PSW تؤخذ على العنوان 96 .

MC (370) $D_1(B_1),I_2$ SI COP=AF MONITOR CALL

تطلق برنامج انقطاع عندما تكون بتة خاصّة من القناع الموجّه في 1 .

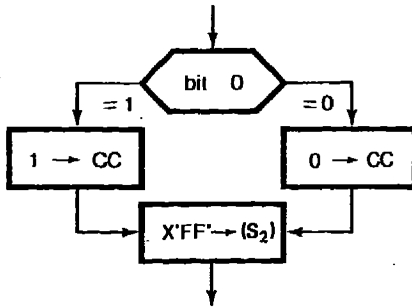
STCK (370) $D_2(B_2)$ S COP=B205 STORE CLOCK



مة الحالية للساعة توضع في كلمة مزدوجة بعنوان S_2 . البتة 31 من ساعة تزداد كل 1,048566 ثانية . ويتم تركيز كود الشرط حسب حالة الساعة .

TS D2(B2) S COP=93 TEST AND SET





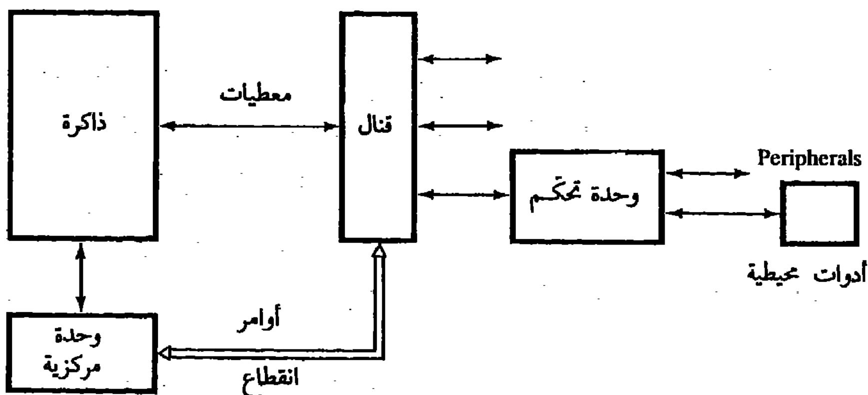
هذه التعليمة تفحص البتة 0 من البايتة بعنوان S2 وبعد ذلك تضع جميع البتات في 1. يتم تركيز CC . لا يمكن قطع هذه التعليمة . وتستعمل بشكل خاص للتحكم بتقاسم المصدر بين عـملين (Processus) و (CROCUS, systemes des exploitation des ordinateurs, Dunod)

2.19 . الإدخال ـ الإخراج

سنعرض هنا للعمليات المهمة لإجراء المداخل والمخارج . بإمكان القارىء ، عند القيام باختباراته ، إجراء إدخال ـ إخراج باستعمال حلقات من فورتران ، مثلاً ، أو بفضل وجود ماكرو تعليمات موجودة على النظام الذي يعمل عليه . سنعود بعد قراءة العموميات إلى دراسة ماكرو تعليمات الإدخال ـ الإخراج .

1.2.19 . تعريف وأوالية الإدخال ـ الإخراج

عملية الإدخال ـ الإخراج هي عملية نقل المعطيات من الذاكرة إلى الأدوات المحيطية وبالعكس وتتم بأمر من الوحدة المركزية تحت مراقبة وتنفيذ القنال .



عند إطلاق العملية فإنها تدور دون تدخل الوحدة المركزية . يظهر القنال وكأنه مُعالج مُستقل ومُخصَّص لتبادل المعطيات بين الذاكرة والجهاز المحيطي . وبشكل عام ،

يوضع البرنامج الذي طلب الإدخال / الإخراج في الانتظار حتى إنتهاء عملية الإدخال / الإخراج . وهذا يعني أن تنفيذه معلّـق خلال مدة الإدخال / الإخراج . وهو يفقد مصادر الوحدة المركزية التي يُمكن أن تُخصّص إلى برامج أخرى مُنتظِرة التنفيذ . بعد إنتهاء عملية الإدخال - الإخراج - وهذا ما يتم إعلام النظام به بواسطة الإنقطاع - سيكون بإمكان البرنامج المقطوع أن يُعاود العمل ، وسيوضع في سجل البرامج التي تنتظر مصادر الوحدة المركزية . هنا يدخل موضوع المزامنة المفروض من الإدخال - الإخراج . يتم تأمين هذا التنظيم والإدارة بواسطة برامج (زُجل) خاصة من نظام التشغيل وهذا هو السبب الذي لأجله لا يستطيع المبرمج أن يُوجِّـه بالكامل عمليات الإدخال - الإخراج الخاصة به . فهو يعطي فقط الإشارات اللازمة لنظام التشغيل ليؤمن حسن تشغيل ودوران برنامجه .

2.2.19 . المعلومات الضرورية لعملية إدخال - إخراج

فلنفكّـر من خلالِ مثلٍ من فورتران . لنفترض عملية كتابة على الطابعة . I و J هي متحولات صحيحة .

```
        WRITE(6,1000) I,J
1000    FORMAT(1X,'I= ',I5,'J= ',I5)
```

إذا كانت قيمة I و J هي على التوالي 4532– و 3 ، نحصل إذاً على :

$I=_{\Delta}-4532_{\Delta}J=_{\Delta\Delta\Delta\Delta\Delta}3$

حيث Δ ترمز إلى الفسحة (البياض) الفارغة .

هذه التعليمة في الإدخال - الإخراج المستوحاة من لغة متطورة تغطي مرحلتين مختلفتين .

- لتحويل المتحولات الصحيحة I و J (ثنائي بفاصلة ثابتة) إلى سمات قابلة للطباعة .

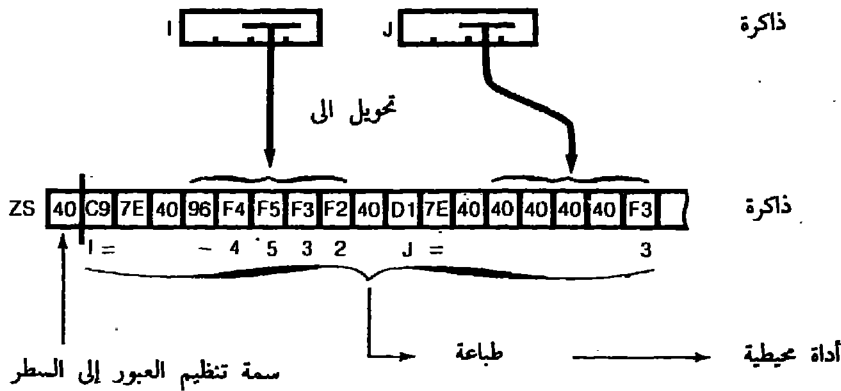
نؤمّـن عملية الإدخال - الإخراج ، أي تبادل المعطيات .

المخطط اللاحق يُوجز العمليات .

النسق FORMAT يُمثّـل إذاً القناع الذي تكلمنا عنه عند دراسة تعليمات الطباعة . المرحلة 1 تتم تحت تحكّـم البرنامج ، المرحلة 2 تقع على عاتق القنال .

نلاحظ إذاً أنه من الضروري معرفة :

- نوع الأداة المحيطية (رقم الوحدة المنطقية ، بلغة فورتران) ،
- العنوان ZS للمنطقة المطلوب طباعتها .



- طول ZS بالبايتات ،
- نوع الأمر (READ أو WRITE) .

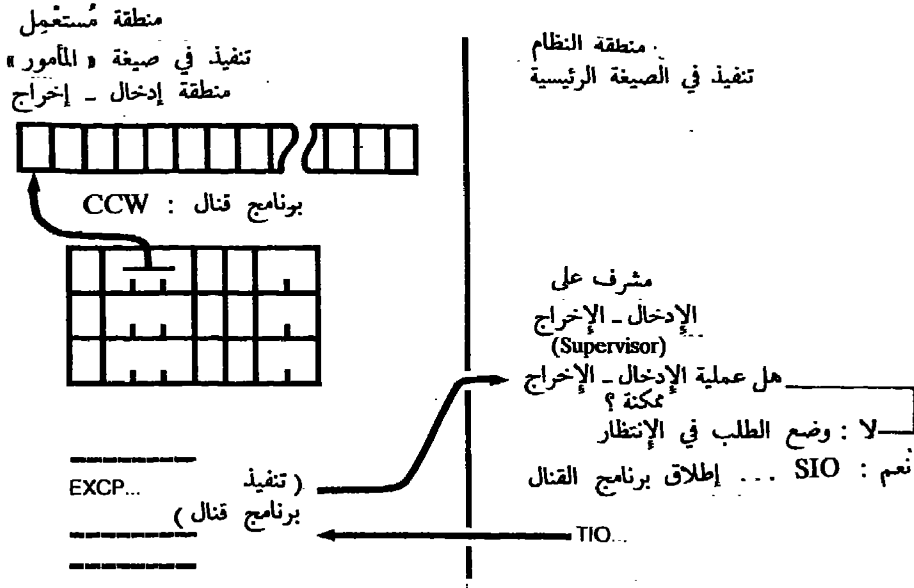
هذه المعلومات إضافة إلى معلومات أخرى ، لأن عمليات الإدخال - الإخراج هي في الواقع أكثر تعقيداً ، يتم وضعها في كلمة مزدوجة للتحكم بالقنال تُدعى CCW (Channel command word : كلمة أمر للقنال) .

يلعب القنال دور الحاسب لأنه قابل للبرمجة . ستُدعى « برنامج قنال » أو « برنامج وحدة تبادل » ، مجموعة الكلمات CCW المكوّنة من أوامر متتالية تتحكّم بالمحيط .

الأدوات المحيطية هي عبارة عن مصادر قابلة للتقاسم والتوزيع بين عدة مُستعْمِلين . يصبح إذاً من الضروري معالجة النزاعات التي قد تولد من جراء طلبات مُتزامنة لنفس المصدر . لهذا السبب فإن مسؤولية إطلاق برنامج القنال تقع على عاتق نظام التشغيل الذي سيتحقّق من توفّر القنال والوحدة المحيطية . وبشكل آخر ، بإمكانه أن يأخذ بعض القرارات في حالة حدوث تنفيذ خاطىء لعملية الإدخال - الإخراج . الكلمة - المزدوجة ذات العنوان 40 ، بالنظام السادس عشري ، والتي تُدعى CSW (Channel status word) ، تعطي بعض المعلومات حول دوران ومحاولة إطلاق الإدخال - الإخراج . المخطّط الوارد على الصفحة التالية يقوم ببعض عمليات الربط بين مختلف العناصر الضرورية للإدخال - الإخراج .

3.2.19 . إدخال - إخراج في المستوى المنطقي

إنّ تنفيذ عملية إدخال - إخراج بالمستوى الفيزيائي هو أمر معقد . كتابة CCW تتطلب معرفة واضحة بالمحيطات التي نعمل عليها . ونعرف أنه في أغلب الوقت تكون



عمليات الإدخال ـ الإخراج على المحيطات البطيئة مؤجّلة . عندما يقوم المستعمل بتعريف سجل طباعة (حالة (6,...) WRITE بلغة فورتران) ، فإن هذا السجل هو أولاً مكتوب على قرص مغناطيسي وبعد ذلك ، بواسطة برنامج خاص ، يُؤخذ لإجراء طباعة نهائية . وفي المجموع فإن رقم الوحدة المنطقي ، يُناسب أولاً فيزيائياً سجلّ قرص مغناطيسي وبعد ذلك سجل الطابعة . هذه العملية ، التي تحاول تبسيط إدارة المصادر المركزية والمحيطية ، تؤدي إلى زيادة الصعوبة في تنفيذ عملية الإدخال ـ الإخراج الفيزيائية . من جهة أخرى ، فإن تنظيم عملية إدخال ـ إخراج يُؤدي إلى درء (Bufferization) لمناطق إدخال ـ إخراج . نعرف أيضاً أنه يوجد عدة تنظيمات نموذجية للسجلات وعدة طرق للبلوغ . هذه الشروط تفرض على المُستعمل بأن يأمن بالكامل لنظام إدارة عمليات الإدخال ـ الإخراج . للقيام بذلك يجب عليه وصف المتغيرات الوسيطة المفيدة بواسطة توجيه من نوع DCB (DATA CONTROL BLOCK) . وهو سيوكل عملية الإدخال ـ الإخراج الخاصة به للنظام بواسطة ماكرو تعليمة خاصة (PUT GET ,...) حسب نوع تنظيم السجل الخاص به . هذه الأخيرة هي موضحة في الوثائق OS/VS2 MVS (DATA Management Macro Instructions) . يقوم النظام بتوليد الكلمات CCW لنفسه ونداء المشرف الضروري . العملية الأولى للإدخال ـ الإخراج ستكون مسبوقة بفتح للسجل (ماكرو OPEN) والأخيرة ستكون متبوعة بإغلاق للسجل (ماكرو CLOSE) يسمح بتفريغ الداريء (Buffer) الأخير . المثل التالي يُوضح ، بإشراف النظام OS ، عملية قراءة بطاقة مثقوبة وكتابة على الطابعة .

```
         OPEN    (CARTE,(INPUT))
         OPEN    (IMP,(OUTPUT))
         ----    ----
         GET     CARTE,ZENTREE
         ----    ----
         PUT     IMP,ZSORTIE
         ----    ----
         CLOSE   CARTE
         CLOSE   IMP
         ----    ----
CARTE    DCB     DDNAME=ENTREE,DSORG=PS,LRECL=80,BLKSIZE=400,MACRF=(GM),
                 RECFM=FB,EODAD=SUITE
IMP      DCB     DDNAME=SORTIE,DSORG=PS,LRECL=133,BLKSIZE=665,MACRF=(PM),
                 RECFM=FBA
ZENTREE  DS      CL'80'
ZSORTIE  DC      133C' '
         ----    ----
```

20 . الأوامر المتعلقة بالعنونة وتركيبة البرنامج

سنقوم بجمع الأوامر (التوجيهات) المستعملة عند بداية ونهاية البرنامج ، التي تسمح بإعداد عداد المواقع ، وتعريف المراصف القاعدية أو تغيير وتقطيع البرامج .

1.20 . تعريف وشحن مراصف القاعدة

لقد عرفنا العنونة القاعدية (فقرة 2.3) وعرضنا مثلاً على تأويل تعليمة من هذا النوع (فقرة 3.3.6) من الضروري العودة الآن بشكل أكثر تنظيماً لهذه المسألة :

إهتمامات المبرمج الأولى هي :

1 ـ تحديد واحد أو عدة عناوين قاعدية .

2 ـ حجز واحد أو عدة مراصف سيتمّ استعمالها كمراصف قاعدية .

3 ـ شحن هذه المراصف بالعناوين المناسبة .

النقطتان الأوليان تتعلقان بمرحلة التأويل ، والنقطة الثالثة تتعلّـق بمرحلة التنفيذ ولا يمكن أن تُحلَّ بشكل نهائي عند التأويل لأن العنوان الفعلي لخزن البرنامج في الذاكرة لن يكون معروفاً إلا في لحظة الشحن .

أ ـ USING

هو الأمر الذي يسمح للمؤول بتحديد مراصف القاعدة وحساب الإزاحة المطلوبة لعنوان محدَّد رمزياً (قاعدة ضمنية ، فقرة 2.9) . وشكله هو التالي :

USING Ad. base, numero des registres de base

USING عنوان قاعدة ، رقم مرصف القاعدة

«Ad. Base» هو تعبير مطلق أو قابل للنقل يعتبره المؤول عنواناً قاعدياً . هذا الأمر لا يُولِّـد أية تعليمة ولذلك فهو لا يزيد من قيمة عداد المواقع . وهو يختفي من البرنامج المُؤوَّل .

مثلاً :

```
(1)  USING  ADBASE,12
(2)  USING  ADBASE,12,11,10
(3)  USING  *,15
```

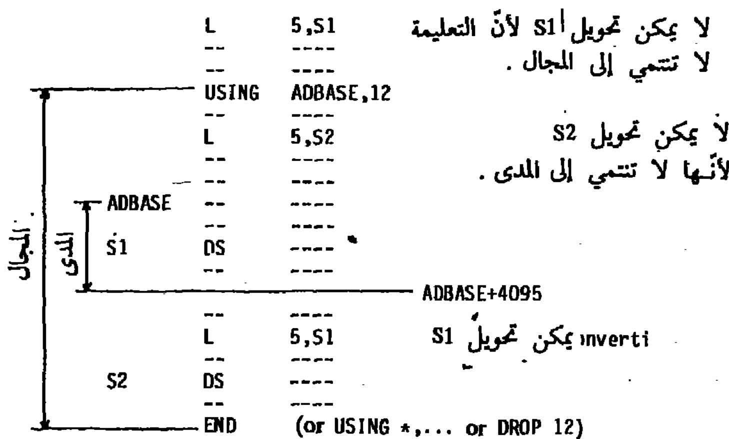
الإزاحة هي كمية مكوَّدة من 12 بتة لا تزيد عن 4095 . وبالتالي ، فإن مدى مرصف القاعدة 12 سيمتد من ADBASE إلى ADBASE + 4095 . عندما يزيد البرنامج عن 4096 بايتة يجب إستعمال الشكل (2) أو عدة أوامر USING لتحقيق العنونة . في الشكل (2) يفترض المؤول أنّ المرصف 12 يحتوي على القيمة ADBASE ، والمرصف 11 القيمة ADBASE+4096 والمرصف 10 القيمة ADBASE + 8192 . في الشكل (3) يفترض المؤول إن العنوان القاعدي هو القيمة الحالية لعداد المواقع .

قواعد الإستعمال

لنميِّز « مدى » المرصف القاعدي من الحقل المُغطى بواسطة تعليمة USING .

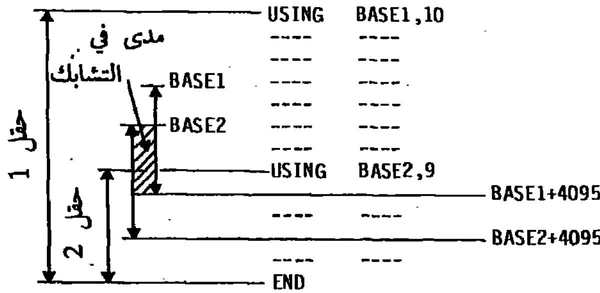
مدى المرصف القاعدي لا يتعلَّق سوى بالعنوان القاعدي المذكور في الأمر وليس بموقع USING . ويمتد من ADBASE إلى ADBASE+4095 . هذا يعني إن جميع الرموز التي تنتمي إلى المنطقة يمكن أن تعنوَن بناء على انتماء التعليمات التي تُرجع إليها إلى « الحقل » .

الحقل USING يمتد من الأمر (التوجيه) USING حتى نهاية (END) الزجلة .
الأمر الآخر USING يُحدِّد نفس المرصف أو يضع الأمر DROP النهاية للحقل السابق .
المثل التالي يوضح ذلك .



حالة استعمال عدة أوامر USING

عندما يتشابك مدى عدة مراصف ، فإن المؤول يحدد بشكل جلي العناوين الرمزية المشتركة لكلا المديين باختيار عنوان قاعدي ذلك الذي ينتج أصغر إزاحة . إذا كانت العناوين القاعدية متشابهة (BASE1 وBASE2 هي ذاتها) ، فهو يختار رقم المرصف الأكبر . إذا كانت العناوين مختلفة ولكن المراصف متشابهة فإن الأمر الثاني USING يقطع مدى الأول



ب ـ شحن مراصف القاعدة

يتوجّه الأمر USING إلى مرحلة التأويل (assembling) . يجب على المبرمج أن يتوقع تعليمة تقوم ، عند التنفيذ ، بتخزين المراصف القاعدية بالعناوين الفعلية الضرورية . هذه العناوين لا يمكن أن تكون معروفة في لحظة التأويل (assembling) لأنها تتعلّق بنقطة الشحن (فقرة 4.6) . المشكلة هي إذاً في كيفية معرفة طريقة استرجاع هذه العناوين . نستعمل لذلك تقنيتين : الطريقة الأولى تستعمل حالة خاصة في استعمال BALR : حيث R_2 هو المرصف 0 (فقرة 4.12) . هكذا فمن الممكن كتابة :

```
BALR    12,0
USING   *,12
```

يُخزِّن عنوان التعليمة BALR زائد 2 (طول التعليمة) في المرصف 12 وهذا العنوان (*) يُحدِّد كقاعدة .

الطريقة الأخرى تقوم على إستعمال إتفاق عادي من النظام OS (فقرة 5.21) بموجبه يُخزِّن النظام في المرصف 15 عنوان نقطة الدخول إلى البرنامج الذي ينتقل التحكّم إليه . هذه هي طريقتنا المفضلة . سنختار كعنوان قاعدي عنوان بداية (نقطة الدخول) إلى البرنامج .

```
DEBUT   CSECT
        -----
        USING   DEBUT,12
        LR      12,15
```

وبالتالي ، وحدها التعليمات التي لا تستعمل عناوين رمزية يُمكن أن تظهر قبل شحن المرصف القاعدي .

ج - DROP

التوجيه أو الأمر DROP $R_1, R_2, ... R_n$ يُشير إلى المؤول لكي لا يستعمل المراصف $R_1, R_2, ... R_n$ كمراصف قاعدية .

2.20 . تقطيع البرامج

كل برنامج مهم يجب أن يكون مقطعاً ، أي مقسماً إلى قطع (زجل module) مستقلة . هذا ما يؤمن لنا بعض الاهتمامات : تبسيط البرامج وتنقيص طول المهام ، إعطاء البرنامج كاملاً تركيبة زجلية تسمح بتسهيل عملية تعديل البرنامج ، تسهيل عمل الفريق (العمل الجماعي)... ونحصل على ذلك بتقسيم البرنامج إلى عدة أقسام - مصدر ، باستعمال الإمكانيات التي تضعها البرامج الثانوية بتصرفنا (أنظر الفصل 21) ، وباستعمال أوامر (توجيهات) التقسيم .

قسم مهم من عمل المؤول يقوم على ربط الرموز الموجودة في الزجل (الأقسام) بعناوين محدّدة على شكل قاعدة ، مؤشر وإزاحة . ينتهي المؤول من العمل عندما يلتقي الأمر END الذي يشير إلى نهاية الزجلة . تتألف الزجلة المصدر من مجموعة من التعليمات المؤولة في مرةٍ واحدة .

1.2.20 . رموز داخلية ، رموز خارجية

يمكن تصنيف الرموز التي يلتقيها المؤول في زجلة مصدرية ، في عدة طبقات .

1 - الرموز المطلقة .

2 - الرموز المنقولة التي تظهر في منطقة الوسم . وهي تسمح عادة ببلوغ تعليمة أو معطى ما . ولا يمكنها أن تظهر إلا مرّة واحدة في منطقة الوسم خوفاً من التعريف المزدوج . كما أنّـها داخلية ضمن زجلة المنبع ويقوم المؤول بربطها بعنوان على شكل قاعدة وإزاحة . ويقوم بتخزينها في جدول الرموز المنقولة (المترجمة) .

3 - الرموز التي تظهر في منطقة الوسم ولكن من النوع « نقاط الدخول » . وتنتمي إلى زجلة المصدر ولكنها قد تكون قابلة للتسمية بواسطة أسماء من خارج هذه الزجلة . من الممكن تصنيفها في طبقتين : طبقة الرموز المستعملة . في تسمية التعليمات ، وطبقة تلك التي تستعمل لتسمية مناطق المعطيات . يقوم المؤوّل بتخزينها في جدول

الرموز الخارجية ESD (External Symbol Dictionary) حتى لو كانت داخلية في زجلة المصدر . رمز واحد على الأقل ينتمي إلى الفئة الأولى : الرمز الذي يشير إلى التعليمة الأولى للتنفيذ . إذا كان هذا الأمر غائباً فإن المؤوّل يختار كنقطة دخول عنوان التعليمة الأولى من البرنامج ويُخزِّنه في ESD . يجب تعداد الرموز من النوع نقاط الدخول في الأمر ... ENTRY SYMB1, SYMB2, إذا لم تكن معتبرة كنقاط دخول إذا كانت مستعملة لتسمية القطعة (الزجلة) .

4 ـ الرموز التي تظهر في زجلة منطقة العوامل ولكن غير الموجودة في منطقة الوسم . هذه الرموز تنتمي إلى زجل مصدرية أخرى ولا يستطيع المؤول أن يربط عنواناً بها ؛ وهو يعهد بهذه المهمّة إلى مُنقِّح الأربطة (link editor) أو إلى الشاحن ، وذلك بتخزينها في ESD . تُعتبر هذه الرموز خارجية بالنسبة لزجلة المصدر . إنّها عبارة عن نقاط دخول إلى زجل أخرى وإذاً فهي تنتمي إلى إحدى الطبقتين المذكورتين في 3 . ويجب أن يكون مصرّحاً عنها وكأنها خارجية بواسطة الأمر ,EXTRN SYMB1 ... SYMB2 إلاّ إذا كانت عبارة عن أسماء برامج ثانوية مصرّحاً عنها في ثابتة بعنوان من النوع V .

2.2.20 . أوامر التقسيم

هذه الأوامر تشير إلى بداية أو نهاية قسم من زجلة المصدر .

END [(مترجم) تعبير منقول]

يشير إلى نهاية زجلة المصدر . العنوان المناسب للتعبير المنقول يُخزَّن في ESD . إنّه بشكل عام عنوان أول تعليمة للتنفيذ .

```
        CSECT
        ----
ALPHA   ----
        ----
        ----
        END     ALPHA
```

يُعرَّف ALPHA كنقطة دخول إلى البرنامج .

قسم التحكُّم (Control section) هو عبارة عن قطعة منقولة من البرنامج (قابلة للترجمة) . هذا يعني بأنّه يجب أن نربط بها مرصف قاعدة واحداً على الأقل ، مما يجعل هذه الوحدة قابلة للنقل والترجمة بشكلٍ مستقل عن باقي البرامج . وهي تبدأ بحدود كلمة مزدوجة . يمتد قسم التحكّم من بداية القسم حتى إلتقاء قسم آخر .

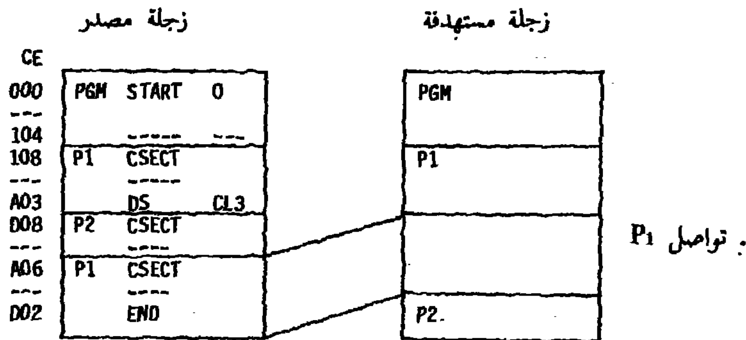
[symbole] START [constante]

[رمز] START [ثابتة]

يقوم بإعداد قسم التحكُّم الأول بزجلة المصدر . الثابتة الاختيارية تسمح بإعطاء قيمة أولية إلى عداد المواقع . يُخزَّن الرمز في ESD .

[Symbol] CSECT

يعرِّف عن قسم التحكُّم أو يؤشر إلى قسم داخلي . الإلتقاء الأول للرمز يشير إلى بداية القسم ، والإلتقاء التالي لنفس الرمز يُشير الى مواصلة القسم . يعمل المؤوّل قسماً بعد قسم : مختلف قطع القسم تكون موجودة متحدة في نفس الزجلة المستهدفة (object module) ، هكذا في المثل التالي ، يتمّ تأويل تواصل P1قبل P2 . من هنا نحصل على قاعدة كيفية تطوّر CE .



تُخزَّن الرموز PGM ، P1 وP2 في الجدول ESD . وهي تُمثّل نقاط الدخول . نشير إلى أن جميع أقسام التحكُّم يجب أن تعرَّف بواسطة رمز ما عدا واحداً . يمكن أن يُعرَّف بواسطة إسم أبيض . يجب على كل قسم ، وهذا موجود في التعريف ، أن يتمتع بمرصف قاعدة . ويُعرِّف المؤول العناوين الفيزيائية للقسم باستعمال هذا المرصف القاعدي الذي يجب أن يُشحن مع قيمة العنوان المناسب . يمكن لقسم التحكُّم أن يبدأ على الشكل التالي :

```
[symbole] CSECT          ( أو START للأول )
          BALR   RBASE,0 ( حيث RBASE هو المرصف القاعدي )
          USING  *,RBASE
```

سنعرض عليكم حلاً آخر لشحن المرصف القاعدي في الفصل 21 .

القسم الوهمي (dummy section) هو عبارة عن قسم مستعمل فقط لوصف المعطيات دون حجز لمواقع لها في الذاكرة ويسمح إذاً بتعريف رموز دون ربطها بعناوين في لحظة كتابة القسم الوهمي . المثل التالي سيوضح ذلك :

لنفترض البرنامج التالي الذي يستعمل المنطقتين Z1 وZ2 المنفصلتين فيزيائياً مع أنّهما بتركيبة متشابهة . سنقوم بتعريف التركيبة المشتركة في تركيبة وهمية تدعى ENREG وسنُطبِّقها على Z1 وZ2 عندما يصبح ذلك ضرورياً .

```
Z1       DS      CL80
Z2       DS      CL80       حجز المناطق
--       --      ----
         USING   ENREG,4
         L       4,=A(Z1)   تعريف العنونة بالنسبة للقسم الوهمي
         --      ----
         --      ----       تطبق تركيبة القسم الوهمي على Z1
         --      ----
         L       4,=A(Z2)
         --      ----
         --      ----       تطبق تركيبة القسم الوهمي على Z2
         --      ----
ENREG    DSECT
NUMERO   DS      CL4
MONTANT  DS      CL10
NOM      DS      CL20
ADRESSE  DS      CL46
```

[symbole] DSECT

يُعرِّف عن بداية أو تواصل القسم الوهمي . عنونة القسم يمكن أن تتم بفضل وجود الرمز الموجود قبل DSECT أو بفضل وجود أي رمز في الوصف . يُوضع عدّاد الرموز دائماً في صفر عند بداية DSECT . يُخزَّن الرمز في ESD . من هنا نلاحظ البساطة الناتجة عن هذا المفهوم . والبرمجة ستكون مُبسَّطة ومن هنا ينتج إقتصاد في استعمال الرموز .

القسم المشترك يسمح لعدة زجل مصدر ، مؤولة بشكل منفصل ولكن متّحدة فيما بينها بواسطة منقح الأربطة ، أن تتقاسم نفس منطقة التنفيذ . سنستعمل هذه المنطقة :

- لإيصال المعطيات بين زجل المصدر (فورتران ومؤوّل مثلاً) ،
- كمنطقة عمل مؤقتة لإحدى الزجل بشرط ألّا تُستعمل في نفس الوقت .

عند المُعالجة بالمؤوّل سيتم حجز موقع لكل زجلة ، ولكن عند المُعالجة بواسطة مُنقِّح الأربطة فإن المناطق المشتركة ستتحد ، وفقط ستحفظ المنطقة ذات الحجم الأكبر .

[رمز] [Symbol] COM

تعرِّف عن منطقة مشتركة . يسمح النظام OS بوسم المناطق ولكن النظام DOS لا يسمح بذلك (لا يوجد رموز) . من الضروري ، في كل زجلة مصدر ، أن يتم

إجراء عنونة بشكل شبيه بما جرى في DSECT . يوضع عداد المراكز في صفر عند بداية القسم .

3.2.20 . تنقيح الأربطة (link edition)

الفقرات السابقة تسمح لنا بفهم وبشكل أفضل عمل مُنقّح الأربطة والشاحن (loader) .

مع الزجلة المستهدفة ، يقدم المؤول إلى مُنقّح الأربطة جدولاً ESD لكل زجلة مصدر . نجد في الجدول ESD أسماء الرموز من الفئتين 3 و4 (فقرة 2.2.20) . في كل رمز نجد كود العملية من نوع الأمر المرتبط بها . إذا كان الرمز من نوع نقطة الدخول ، فإن عنوانه هو في الزجلة المشار إليها . بالنسبة للزجلة المصدر المذكورة في الفقرة 4.2.20، فإن الجدول ESD يكون على الشكل التالي :

```
EXTERNAL SYMBOL DICTIONARY

SYMBOL    TYPE  ID    ACDR    LENGTH LDID
          PC    0001  000000  00001C
ALPHA     ER    0002
P1        SD    0003  000020  00000C
DEBUT     ER    0004
SP        ER    0005
```

يكوّد نوع الرمز على الشكل التالي :

كود	مناسب للأمر
PC	START ou CSECT بدون وسم
SD	START ou CSECT مع وسم
DM	COM
XD	DXD ou DSECT خارجي، (1)
LD	ENTRY
ER	EXTRN DC V(...) أو ثابتة بعنوان
WX	WXTRN (2)

في مقابل هذه المعلومات المرتبطة بكل زجلة ، فإن منقّح الأربطة يقوم بالإجابة على الطلبات الخارجية ، أي يقوم بإجراء التناسب بين الأسماء الموجودة في مختلف ESD . وإذا لم يكن بإمكان المُنقّح أن يحلّ مشكلة الطلبات الخارجية بسبب جدول الزجل ESD المطلوب ربطها ، فهو يقوم بعملية بحث منتظمة في المكتبات التي يقدر على بلوغها .

---

(1) CXD ، DXD ، DSECT الخارجية هي غير مشروحة في هذا الكتاب .
(2) WXTRN تقوم بملء نفس الدور الخاص بـ EXTRN . في ما يتعلّق بالسماح لمنقّح الأربطة بالبحث الأوتوماتيكي عن الرموز بداخل المكتبة ، فإنّ WXTRN تمنع هذا البحث .

4.2.20 . الشحن (loading)

يقوم الشحن على خزن البرنامج في الذاكرة بدءاً من عنوان مُحدَّد . كما رأينا في الفقرة 2.3 ، العناوين المنقولة لا يجب أن تتعدَّل خلال هذه العملية . والأمر ليس كذلك بالنسبة لثوابت العنوان . يقوم الشاحن بخزن العناوين الفعلية للمتأثرات المطلوبة في الذاكرة .

يجب على المؤول أن يرسل إلى الشاحن مواقع المناطق المطلوب إعادة حسابها . يستعمل لهذا الهدف RLD (Relocation Dictionary) حيث تتواجد عناوين ثوابت العنوان . الجدول ESD في المثل أعلاه هو موجود في الفقرة 3.2.20 . نذكّر بأن DC V (SYMB) يعادل :

```
EXTRN SYMB
DC    A(SYMB)
```

يُحتفظ باستعمال ثوابت العنوان من النوع V للتعريف عن عنوان تفريع (إسم قسم ، إسم برنامج ثانوي . . .) الرمز SYMB يُخزَّن في ESD . ويقوم المؤول بتصفير الثابتة .

```
LOC    OBJECT CODE   ADDR1 ADDR2  STMT   SOURCE STATEMENT
000000                              1           START 0
                           00000    2           USING *,12
                                    3           EXTRN  ALPHA
000000 5830 C010     00010          4  DEBUT    L      3,=A(ALPHA)
000004 5850 C014     00014          5           L      5,=A(BETA)
000008 FFFFFFFF                     6  ZONE     DC     F'-1'
000020                              7  P1       CSECT
                           00020    8           USING  *,11
000020 5850 C018     00018          9  BETA     L      5,=V(SP)
000024 00000000                    10  ADR      DC     A(DEBUT)
000028 00000000                    11           DC     V(DEBUT)
                                   12           END
000010 00000000                    13                  =A(ALPHA)
000014 00000020                    14                  =A(BETA)
000018 00000000                    15                  =V(SP)
```

RELOCATION DICTIONARY

POS.ID	REL.ID	FLAGS	ADDRESS
0001	0002	0C	000010
0001	0003	0C	000014
0001	0005	1C	000018
0003	0001	0C	000024
0003	0004	1C	000028

سنفحص في المخطط التالي كيفية تطوّر القيمة المأخوذة من قبل ثابتة عنوان من التأويل إلى الشحن :

زجل مصدر

```
PROG1 START  0
      EXTRN  SYMB
      USING  PROG1,12
      -----  -----
      L      2,ASYMB
      L      2,0(2)
      -----  -----
ASYMB DC     A(SYMB)
      -----  -----
      END
```

```
PROG2 START  0
      ENTRY  SYMB
      -----  -----
      -----  -----
SYMB  DC     F'-1'
      -----  -----
      END
```

زجل مؤولة

```
CE
000
000
000
---   --------
100   5820C200
104   58220000
---   --------
200   00000000
---   --------

000
000
---   --------
---   --------
124   FFFFFFFF
---   --------
```

زُجل بعد المُعالجة بمنقّح الأربطة والشحن

```
--------
5820C200
58220000
--------
00077400
--------

--------
--------
FFFFFFFF
--------
```

00077400
+
عنوان فعلي

5.2.20 . الاتصال بين أقسام نفس الزجلة المصدر

لنأخذ المثل التالي :

```
  LOC   OBJECT CODE     ADCR1 ADDR2  STMT    SOURCE STATEMENT
000000                                1 P1        CSECT
                                      2 *         ---
                              00000   3           USING P1,12
                                      4 *         ---
C00000 0000 0000        00000         5           L      3,SYMB2
        *** ERROR ***
                                      6 *         ---
000004 5840 C010        00010         7           L      4,=A(SYMB2)
C00008 5844 0000        00000         8           L      4,0(4)
                                      9 *         ---
C0000C 0C000001                      10 SYMB1     DC     F'1'
                                     11 *         ---
000018                               12 P2        CSECT
                                     13 *         ---
                              00018  14           USING P2,11
                                     15 *         ---
000018 5830 C00C        0C00C        16           L      3,SYMB1
                                     17 *         ---
00001C FFFFFFFF                      18 SYMB2     DC     F'-1'
                                     19 *         ---
                                     20           END
000010 0000001C                      21                  =A(SYMB2)
```

ولنعرض المشاكل التي يفرضها الاتصال بين قسمين عند إجراء مرحلتين من التأويل والتنفيذ .

1 ـ عند التأويل فإن أي مشكلة خاصة لن تواجهنا . ينتمي القسمان إلى نفس زجلة المصدر ويمكن أن يقوم المؤول بإجراء شروط العنونة لتجميع الرموز الداخلية بشرط أن توافق القواعد العائدة إلى USING . هكذا ، فتأويل السطر الخامس لا يمكن أن يتم لأن هذه التعليمة لا تنتمي أبداً إلى حقل USING P2,11 . في القسم P1 ،

نستطيع بلوغ SYMB2 باستعمال ثابتة العنوان A(SYMB2) التي يقوم الشاحن بإعدادها بشكل مناسب . وفي المقابل ، فإن التعليمة '16 L 3,SYMB1 يمكن أن تكون مؤولة .

2 ـ عند التنفيذ ، تكون المشكلة مختلفة : التعليمة L 3, SYMB1 هل ستسمح بالبلوغ إلى SYMB1 ؟

قد يسمح لنا التأويل المناسب للتعليمة بهذا الافتراض . هكذا فعملياً هذه التعليمة تسمح عند التنفيذ ، ببلوغ SYMB1 بشرط أن تكون القاعدة 12 المُعنونة SYMB1 تحتوي على العنوان P1 المناسب . ولكن لا شيء مؤكداً ، في مثل مُعاكس ، يكفي أن يكون القسم P2 مُنفّذاً قبل القسم P1 كي لا تكون القاعدة 12مشحونة بشكلٍ مناسب . إضافة لذلك ، فإن أي مراجعة من هذه الطبيعة تناقض تعريف قسم التحكّم . وبالتالي فإنّنا سنراجع SYMB1 في P2 بفضل وجود ثابتة العنوان .

يظهر إذاً وبوضح أنّ الأقسام يجب أن تُعتبر كوحدات مُستقلة في نفس الوقت الذي تكون فيه الزجل المصدرية منفصلة عند التأويل . الاتصال الرمزي بين الأقسام سيتم دائماً بواسطة ثوابت العنوان . هذه التقنية تسمح بتفادي العقبة المُثارة أعلاه وتسمح بدون مشكلة بتوزيع الأقسام في مختلف زجل المصدر .

وبإيجاز ، فإن تفريع القسم سيتم بواسطة :

```
L     R,=V(P1)   ( = A (P1) أو )
BR    R          حيث R هو مرصف عام
```

R هو مرصف عام ، بشكل عام المرصف 15 حسب إتفاقات الربط المعروضة في الفقرة 4.21 .

بلوغ الرمز يتم بواسطة :

```
L     R,=A(SYMB)
L     R,0(R)
```

6.2.20 . ختام حول التقسيم

يعطي التقسيم وسيلة لتجزئة زجلة المصدر إلى زجل مُستقلة . عند إجراء التقسيم فإن كل شيء يجري كما لو كانت الزجل المصدرية مترابطة .

نحرص على عدم بلوغ ، في نفس القسم ، رموز لا تنتمي إلى هذا القسم . وإذا كنا نرغب ببلوغ رموز خارجية فسنستعمل الطريقة المعروضة في الفقرة 4.2.20 ، تاركين إلى الشاحن مهمّة إجراء الوصلة بواسطة ثوابت العنوان .

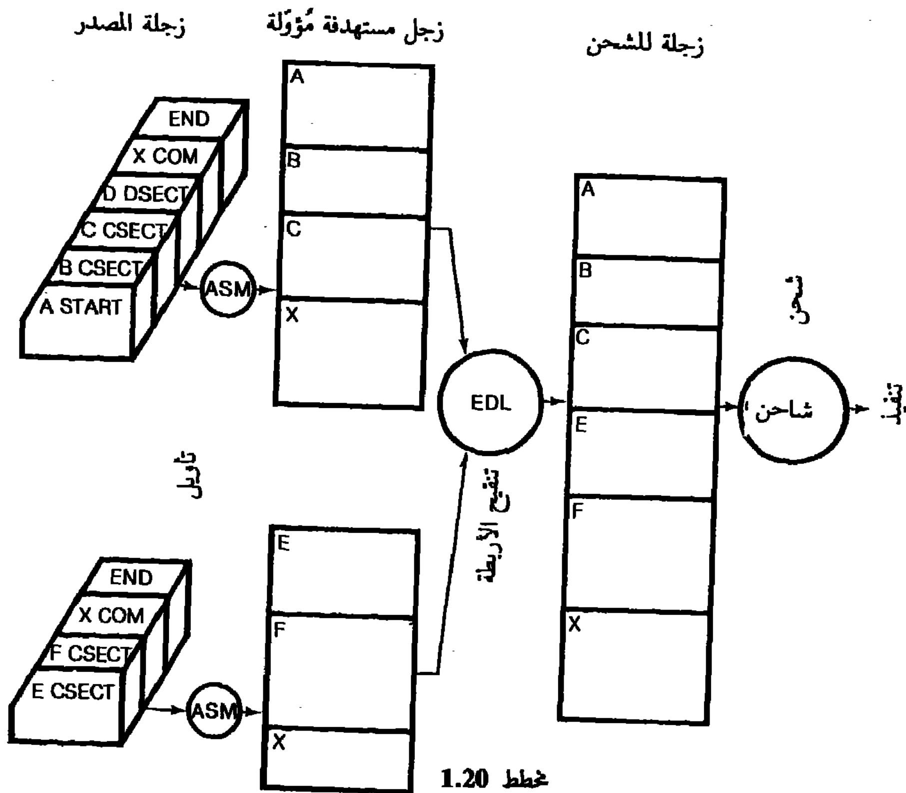
يجب على كلّ قسم أن يحتوي على مرصف قاعدة ، ويجب شحن هذا المرصف ، في لحظة التنفيذ ، بالعنوان المناسب . ستجري دراسة هذه المسألة في الفصل التالي .

بعد أخذ هذه الاحتياطات بعين الاعتبار ، فإن التقسيم يؤدي إلى تحسين كبير في تنظيم المعالجة بالمؤول . وهو يسمح ، عند الحاجة ، « بتفتيت » وبدون مشكلة البرنامج إلى زجل دون أي خوف على الترابط العام .

وبشكل عام فإن الأقسام هي برامج ثانوية . يجب أخذ الاعتناء ، عند الدخول إلى قسم من هذا النوع ، بتخزين مراصف البرنامج المُنادي .

ويُعالج الفصل 21 هذه المشكلة . لا يجب الخلط بين القسم والبرنامج الثانوي اللذين يمثلان مفهومين مختلفين . من الممكن القول أن تقسيم البرنامج هو عبارة عن نقل قسم من العمل الجاري بواسطة المؤول إلى مُنقّح الأربطة والشاحن .

سنلاحظ في المخطّط التالي إختفاء DSECT من الزجلة المؤولة والموقع الوحيد المشغول بواسطة COM في الزجلة المشحونة . المكان المشغول بواسطة القسم المشترك يُعادل الحجم الأكبر بين الاثنين .



مخطط 1.20

3.20 . الأوامر التي تُغيِّر عدّاد المواقع

ORG عبارة عن تعبير منقول أو مطلق . هذا الأمر يؤدي إلى تغيير الازدياد الطبيعي لعدّاد المواقع . وهو يسمح بشكل خاص بإجراء إعادة تعريف أو حجز مكان من الذاكرة . إذا كانت منطقة العناصر (القياسات) فارغة ، فإن ORG يعطي عداد المواقع CE القيمة التي كانت موجودة فيه عند آخر تعديل بواسطة ORG . لا يمكن أن يكون القياس (argument) مبلوغاً في البداية .

```
قيمة العداد
 CE

0C0      TABLE   DC      XL256'40'
0CA              ORG     TABLE+10
                 ---
0D0              ORG
                 ---
```

LTORG عبارة عن أمر بدون قياسات . وهو يشير إلى المكان الذي يجب أن تُؤوَّل فيه الثوابت الحرفية . في غياب هذا الأمر فإن تأويلها سيتم في نهاية أول قسم .

CNOP b, w يُؤدي ، بحكم عدم إجراء أية عملية ، إلى زيادة قيمة عداد المواقع إلى الحد الأقرب لنصف كلمة ، كلمة أو كلمة مزدوجة حسب قيمتين b و w .

CNOP	0,4	بداية كلمة
CNOP	2,4	وسط كلمة
CNOP	0,8	بداية كلمة مزدوجة
CNOP	2,8	النصف كلمة الثاني من كلمة مزدوجة
CNOP	4,8	النصف كلمة الثالث من كلمة مزدوجة
CNOP	6,8	النصف كلمة الرابع من كلمة مزدوجة

4.20 . أوامر التحكُّم باللوائح

ICTL يسمح بتعديل الإطار النموذجي (الأعمدة 1 ، 16 و 71) للتعليمات .

ISEQ يسمح بالتحقّق من الترتيب المتتالي للبطاقات .

COPY يسمح بنسخ قسم من النص المصدر في المكتبة .

EJECT يؤدي إلى ظهور التعليمة التالية في رأس الصفحة التالية من اللائحة . وهو مفيد لتوضيح نصّ البرنامج .

SPACE n يسمح بإدخال عدد n من الأسطر الفارغة في اللائحة .

PRINT [ON, OFF ; GEN, ,NOGEN ; NODATA, ,DATA]

يسمح بالمحافظة على أو بإلغاء اللائحة (Listing) ، توليد الماكرو تعليمات، توليد المعطيات .

«سلسلة» TITLE يسمح بطباعة عنوان من 100 سمة في رأس كل صفحة .
PUNCH, REPRO يسمحان بتثقيب البطاقات .

5.20 . أوامر مُستعملة بإشراف النظام OS فقط

OPSYN يسمح بتعريف مجموعة كود العمليات الخاصة المُرادفة للأكواد IBM .

هذا الأمر يمكن أن يكون مفيداً بشكل خاص لاستبدال كود ـ عملية خاص بماكرو عملية .

من الممكن إذاً تبديل الكود الحرفي BE ، BNE ، . . . . للماكرو حيث الأسماء سيصرَّح عنها بشكل مرادف بسبب وجود OPSYN . هذه الماكرو تعليمات تولِّد كلمة تُخزَّن فيها نتيجة الاختبار الذي يسبق تعليمة التفريع بالشكل V أو F أو O أو N » وبعد ذلك تقوم بالتفريع المناسب باستعمال التعليمة BC أو BCR . هذه السمات V أو F ستكون مرئية في العملية DUMP (دلق) وتسمح بمتابعة أثر تنفيذ البرنامج (Trace) . بالإمكان تمييز مختلف الأسماء المُولَّدة بواسطة SYSNDX& (فقرة 7.2.22) .

بعد مرحلة الإطلاق في العمل ، فإن إلغاء الأوامر (التوجيهات) OPSYN يؤدي إلى تفادي إدخال ماكرو التعليمات والبدء بتنفيذها .

من الممكن أيضاً إستعمال هذا الأمر لجعل بعض التعليمات غير عملية وذلك بجعلها مرادفة للتعليمة NOP (لا عملية) .

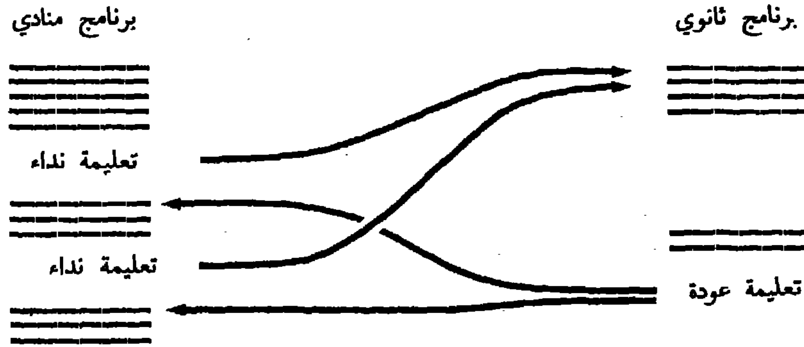
PUSH وPOP . من الممكن عند كتابة البرنامج أن نقوم بشحن مرصف القاعدة بسرعة وأن نستعيد القاعدة القديمة لاحقاً . هذا يمكن أن يتم مثلاً ، عندما تستعمل إحدى ماكرو التعليمات قاعدة شخصية . بعد التبديل ، بواسطة المؤول ، يجري فقدان القاعدة القديمة . يسمح الأمر PUSH بتخزين المراصف وعنوان القاعدة وصيغ الأمر PRINT داخل مكدس (Stack) (1) . POP يُعاود إسترجاع المفهوم القديم بواسطة إستخراج لآخر كلمة مكدّسة .

---

(1) المكدس هو عبارة عن جدول مُنظَّم حسب التقنية « الداخل أخيراً هو الخارج أولاً » .

21 البرامج الثانوية

البـرنـامج الثانوي هو عبارة عن سلسلة من التعليمات التي يتم تنفيذها بطلب من تعليمة نداء (Call) . عندما ينتهي تنفيذ البرنامج الثانوي يعود العمل بالبرنامج المُنادي وبالتعليمة التي تتبع مباشرة تعليمة النداء . المخطط التالي يوضح هذه الأوالية :



كل شيء يجري كما لو كانت تعليمات البرنامج الثانوي داخلة في مكان تعليمة النداء .

بإمكاننا تقسيم البرنامج الى مهام (task) ، كل مهمة يتم حلّـها بواسطة برنامج ثانوي . إعداد البرنامج بكامله يصبح سهلاً ، والأقسام تصبح صغيرة . هذه الأوالية تسمح بتفادي إعادة كتابة التعليمات المتشابهة عندما يجب تنفيذ البرنامج في مختلف مستويات البرنامج المُنادي . وتطرح هذه التقنية مشكلتين :

- تخزين عنوان العودة (العنوان الذي يتبع مباشرة عنوان تعليمة المُناداة) ،
- إنتقال المتغيرات الوسيطة .

مشكلة إنتقال المتغيرات جرت إثارتها في إطار تقسيم البرنامج ولكن البرنامج الثانوي لا يُشكّـل بالضرورة قسم تحكّـم

1.21 ـ البرنامج الثانوي وقسم التحكّم

التقسيم هو عبارة عن عملية تتعلّق بالتأويل ، تنقيح الأربطة والشحن . أمّا مفهوم البرنامج الثانوي فلا يتعلّق سوى بالتنفيذ . مناداة البرنامج الثانوي تؤدي ، عند التنفيذ ، إلى تعديل الدوران المتتالي للتعليمات .

هكذا ، فلا شيء يعترض بأن يكون البرنامج والبرنامج الثانوي تابعين لنفس القسم . ولكن هذا النوع من التنظيم لا يُقدِّم جميع الفوائد التي ننتظرها من البرنامج الثانوي . فهو يربط البرنامج بالبرنامج الثانوي بينما نرغب نحن بجعل البرنامج الثانوي قابلاً للطلب والدعوة من جميع الأقسام أو الزجل . وهو لا يشكل تحسيناً باتجاه تركيبة زجلية . وبالتالي لا يستعمل إلا عندما يكون البرنامج الثانوي مرتبطاً بشكل كبير منطقياً بالبرنامج المُنادي .

في أغلب الأحيان يُفضَّل إستعمال إمكانيات التقسيم : سيشكل البرنامج الثانوي قسماً من البرنامج .. من المحتمل ، منذ لحظة تصوُّر البرنامج الثانوي ، إستعمال هذه الزجلة في مُعالجات أخرى . يُفضل معالجة مشكلة الاتصال بين البرنامج / البرنامج الثانوي كوصلة ببرنامج خارجي تسمح بإمكانية تفكيك عمليات التأويل دون تعديل في الأقسام .

2.21 . تفريع إلى برنامج ثانوي والعودة

مناداة البرنامج الثانوي ليست سوى قطع إلزامي للدوران المتتالي للتعليمات ولكن مع تخزين للعنوان التالي الذي يتبع تعليمة المنادة بشكل يسمح بمعاودة العمل بالبرنامج المقطوع . تتمتع كل مكنة بأوالية خاصة للتفريع مع عودة . يستعمل النظام 360/370 التعليمتين BAL وBALR اللتين رأيناهما في الفصل 12 .

```
BAL    R1,D2(X2,B2)
BALR   R1,R2
```

يكون عنوان العودة مُخزَّناً في المرصف R1 . يكفي إذاً في نهاية البرنامج الثانوي أن نشحن عدّاد البرنامج بالقيمة المخزّنة في R1 بواسطة التعليمة BCR 15,R1 مثلاً . نحصل إذاً على التركيبة التالية :

البرنامج المنادي	البرنامج الثانوي SP

-----	SP -----
L R2,=A(SP) (أو V(SP))	(تخزين المراصف وتعريف القاعدة)
BALR R1,R2 إذا كان SP خارجياً	
-----	-----

- - - - -

- - - - -

(إعادة مضمون المراضف إلى الذاكرة)

BCR 15,R1

- - - - -

إذا كانت BALR موجودة على العنوان ALPHA ، فإن BCR 15,R1 تعيد خزن ALPHA +2 في عداد البرنامج (CO) .

كان بإمكاننا إستعمال BAL بأحد الأشكال التالية :

1°) BAL R1,SP إذا كان SP عبارة عن مرجع داخلي

```
2°) L      R2,=A(SP)        ou =V(SP)
    BAL    R1,DEPLAC(R2)
```

الشكل الذي يسمح ، بواسطة حساب بسيط لِـ DEPLAC ، بالحصول على مداخل متعدّدة في SP .

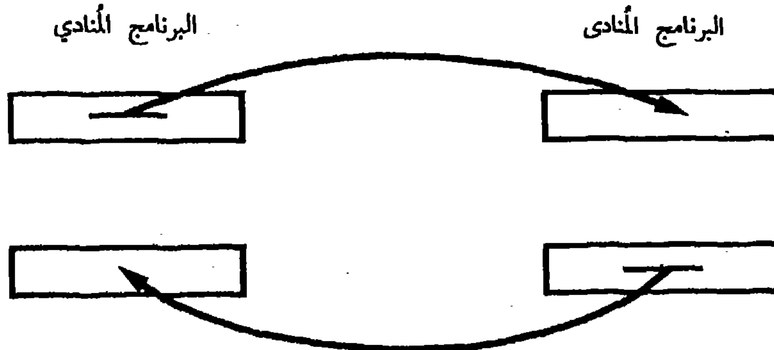
لنلاحظ أنّـه لا يوجد فرق أساسي بين التفريعات إلى .برامج ثانوية خارجية أو داخلية . وحده تعريف ثابتة العنوان الخارجي هو إلزامي في الحالة الأولى .

3.21 . إنتقال المتغيرات الوسيطة

المشكلة الثانية في عملية الاتصال بين البرنامج والبرنامج الثانوي تكمن في عملية تبادل المعطيات . إنّ تقنيات عبور المتغيرات هي متعددة ويمكن للقارىء أن يتصوّر الطريقة الأفضل لمسألته . ولكن من المفيد هنا أن نعرض الطرق العامة التي تساعده على الاختيار . تُستعمل اللغات المتطورة بطريقين أساسيين : لانتقال المتغيرات مباشرة بالقيم والانتقال بالعناوين .

إنتقال المتغيرات حسب القيم

ويكمن في نسخ القيمة المطلوب إرسالها إلى منطقة معروفة من البرنامج المنادى .



هذه المنطقة يمكن أن تكون خلية في الذاكرة مركزية (Local) في البرنامج المُنادى أو مرصفاً. تستعمل هذه التقنية ، مثلاً في لغة فورتران ، لاعادة قيمة إحدى الدوال إلى البرنامج المُنادي .وبشكل عام فإن النتيجة تخزّن في المرصف 0 بواسطة البرنامج المُنادي .

نلاحظ إنه إذا كانت B عبارة عن متحوّلة مركزية من البرنامج المُنادى ، فإن أي تعديل في B لا يؤدي إلى أي تغيير في الخلية A .

وفي لغة المؤول ، يمكن أن تُحلّ مشكلة التبادل بالقيم بواسطة النقل بالمراصف ، حيث يُحدِّد المبرمج طريقة لاستعمال المراصف .

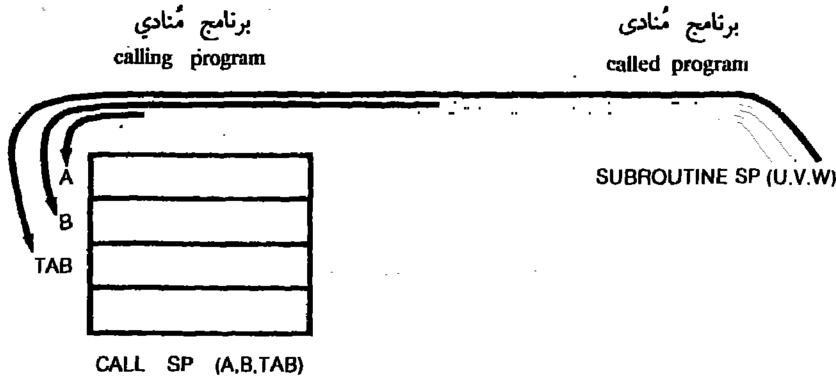
البرنامج المُنادي	البرنامج المُنادى

L 1,A	-------
L 2,B	ST 1,U
تفريع إلى SP au SP	ST 2,V

نشير إلى أن هذه الأوالية هي غير متوافقة مع تبادل الجداول. فعندئذٍ تتطلّب مكاناً كبيراً من الذاكرة . هذه الطريقة هي غير مناسبة إلا عندما يكون عدد المعطيات المطلوب إرسالها قليلاً .

إنتقال المتغيرات بواسطة العناوين

وتكمن هذه الطريقة بإرسال عناوين المتغيرات إلى البرنامج المُنادى . يعمل البرنامج المُنادى إذاً على معطيات البرنامج المُنادي . يبلغ البرنامج المُنادى قِيم المتغيرات بواسطة العنونة غير المباشرة . أي تعديل ، في البرنامج المنادى ، في قيمة منقولة ، معناه تعديل منطقة من البرنامج المُنادي . هذه الطريقة هي نفسها المُستعملة للارسال بواسطة CALL (Call SP name, arguments list) في فورتران . المخطط التالي يوضح لنا عملياً كيف أن متحولات البرنامج المُنادي تصبح مركزية في البرنامج المُنادى .



تُدعى متغيّـرات وهمية الرموز A ، B ، TAB الواردة في تعليمة النداء لأنها تتمتع فعلياً بقيمة معينة في لحظة النداء أو عند العودة .

تُدعى متغيرات شكلية الرموز U ، V ، W من SP التي ليست سوى أسماء تمثّل ، في لحظة النداء ، الرموز A ، B ، TAB من البرنامج المُنادي .

في لغة المؤول بإمكان المبرمج تصوّر عدة حلول لنقل المتغيرات إلى البرنامج المركزي ، فلنذكر البعض منها .

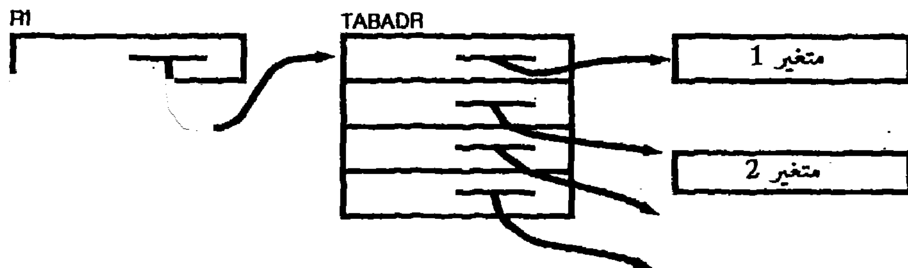
1 ـ نضع المتغيرات في الجدول TAB ونرسل عنوان الجدول بواسطة أحد المراصف .

نداء		برنامج ثانوي
L	R1,=A(TAB)	يتم بلوغ المتغيّر n بواسطة
L	R15,=A(SP)	L R4,DEPLAC(R1)
BALR	R14,R15	أو بالتأشير
		L R4,0(R5,R1)
		وعندئذٍ يوضع المتغير بتصرفه في R4

2 ـ نضع عنوان الجدول TAB مباشرة بعد تعليمة النداء

النداء		برنامج ثانوي
L	R15,=A(SP)	R14 يسمح ببلوغ TAB .
CNOP	2,4 (تراصف)	العودة تتم بواسطة :
BALR	R14,R15	BC 15,4(R14)
DC	A(TAB)	

3 ـ تكون المتغيرات عادة غير متراصة في البرنامج ونُفضّل عادة اعتماد التقنية المستعملة بواسطة المصرّفات . نقوم بإرسال عنوان الجدول الذي يحتوي على عناوين المتغيرات بواسطة أحد المراصف .



نداء		برنامج ثانوي	
L	R1,=A(TABADR)	WORK EQU ...	مرصف عمل
L	R15,=A(SP)		
BALR	R14,R15	L	WORK,0(,R1)
		L	WORK,0(,WORK)
			المتغيّر الأول في WORK
		L	WORK,4(,R1)
		L	WORK,0(,WORK)
			المتغير الثاني في WORK

هذا الحلّ هو المعتمد في لغة فورتران ، ويسمح ، في لغة المؤول ، باستعادة المتغيرات المرسلة بواسطة أحد البرامج فورتران وبالعكس .

نشير هنا إلى الفرق بين المتغيرات المرسلة ومتغيرات العودة ، وهي تنتمي إلى البرنامج المُنادي . كما نفضـل إستعمال مراصف حسب نفس الاتفاقات المستعملة في أنظمة التشغيل (فقرة 4.21) . تسمح التعليمة CALL بإرسال من هذا النوع .

4.21 . إتفاقات الإتصال بين النظام والبرنامج

يبدأ التنفيذ منذ اللحظة التي يتم فيها إعداد عدّاد البرنامج وتخزين عنوان التعليمة الأولى للتنفيذ فيه . يقوم نظام التشغيل بهذه المهمة ، مما يفترض علينا إعتبار كل برنامج مستعمل كبرنامج ثانوي للنظام . من هنا فإن برنامج المستعمل يجب أن يبدأ بتمهيد يتعلّـق بشروط إستعمال المراصف من قِبل النظام .

تسمّـى المراصف 0 ، 1 ، 13 ، 14 و15 مراصف ربط «linkage registers» في وثائق المُصمّم . وتستعمل بواسطة النظام والمصرّفات بشكل نموذجي وهذا هو السبب الذي من أجله يعتمد المستعمل على نفس الاتفاقات في الاتصالات مع البرامج الثانوية الخاصة به . في النظام OS ، يجب على البرنامج الثانوي أن يحمي مراصف المُنادي في منطقة تدعى SAVE AREA ، تنتمي إلى البرنامج المنادي . تحدَّد تركيبة هذه المنطقة على الشكل التالي :

الكلمة	المحتوى
1	تستعمل بواسطة اللغة PL/1
2	عنوان SAVE AREA الداخلي السابقة (الخاصة بالمُنادي) .
3	عنوان SAVE AREA التالية (الخاصة بالمُنادى) .
4	عنوان العودة إلى المُنادي (مرصف 14) .
5	عنوان نقطة الدخول إلى البرنامج (مرصف 15) .
6	مرصف 0 .
7	مرصف 1 .
.....	.....
18	مرصف 12 .

عندما ينقل النظام التحكّم إلى البرنامج :

- يحتوي المرصف 15 على عنوان نقطة الدخول إلى البرنامج . بإمكان البرنامج المُنادى أن يشحن المرصف القاعدي الخاص به بواسطة التعليمة LR REGBASE,15 ، باعتماد نقطة الدخول وكأنها عنوان قاعدي .
- المرصف 14 يحتوي على عنوان العودة .
- المرصف 13 يحتوي على العنوان SAVEAREA للبرنامج المُنادي . نجد هنا شرح إستعمال القاعدة 13 في التعليمة STM 14,12,12(13) الموجودة في جميع التمهيدات للبرامج .
- المرصف 1 يحتوي على عنوان جدول الكلمات التي تحتوي على عناوين المتغيرات الوهمية المنقولة . هذا الإتفاق يُستعمل ، مثلاً ، عندما يطلب برنامج فورتران برنامجاً آخر بلغة المؤوّل .
- المرصف 0 ، يستعمل ، عند العودة ، لإرسال نتيجة إحدى الدوال (مثلاً الدالة FUNCTION في فورتران) .

وبالنتيجة ، ومنذ اللحظة التي يأخذ فيها البرنامج المُنادى التحكم ويعود إلى التنفيذ ، فإنه :

- يُعرِّف المنطقة الخاصة به SAVE AREA ،
- يُخزن مراصف البرنامج المُنادي بواسطة :

STM 14, 12, 12 (13)

في المنطقة SAVE AREA للمنادي

- يعرِّف مرصف قاعدة ويشحن فيه قيمة معينة بواسطة :

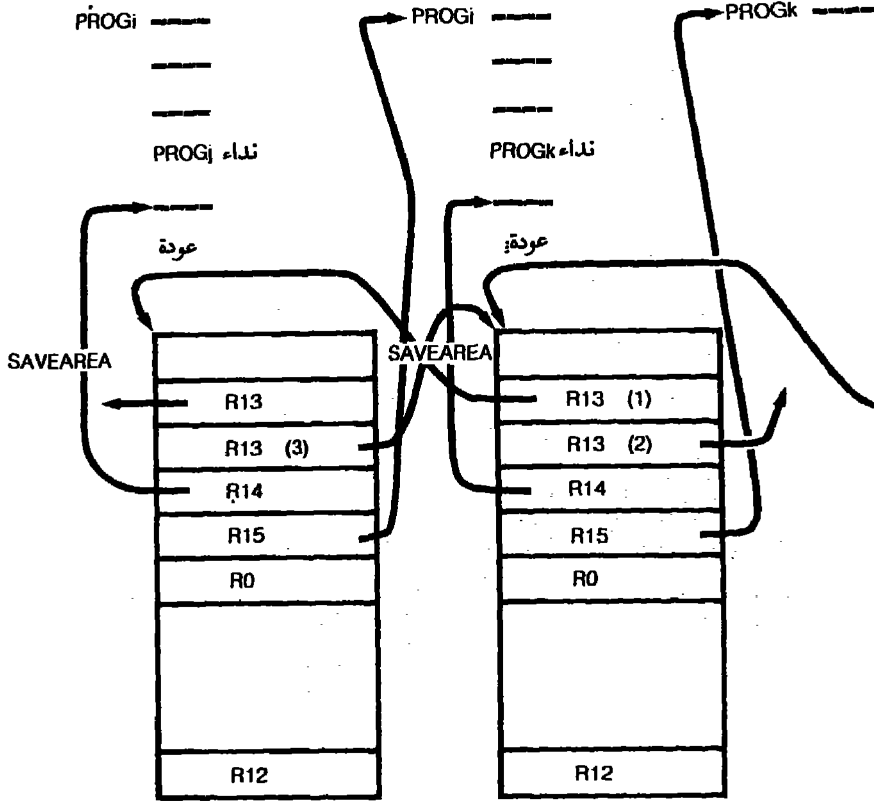
BALR ....., 0 أو LR ....,15

- يقوم بإجراء الوصلة بين المناطق SAVE AREA : ويُخزِّن ، في الكلمة الثانية من المنطقة SAVE AREA الخاصة به عنوان المنطقة الخاصة بالبرنامج المُنادي (مرصف 13) وفي الكلمة الثالثة من المنطقة SAVE AREA الخاصة بالمُنادي ، عنوان المنطقة SAVE AREA الخاصة به .

عند العودة ، فإن البرنامج المُنادى يعيد تخزين مراصف البرنامج المُنادي مما يؤدي إلى العودة بواسطة BR 14 .

بإمكانه إستعمال المرصف 15 لترميم كود العودة .

المخطط التالي يُوضح عملية الربط بين المناطق SAVE AREA .



مخطط 1.21

ملاحظات : إذا كان البرنامج المُنادى ، PROG j مثلاً ، لا ينقل التحكُّم إلى برامج ثانوية أخرى كالبرنامج PROGk ، فلا حاجة لتعريف SAVE AREA لهذا البرنامج . من الواجب إذاً السهر على حماية المرصف 13 الذي يسمح بإعادة مفهوم التنفيذ الى البرنامج المُنادي .

---

(1) يتعلّق ذلك بالمرصف R13 من PROGi

(2) يتعلّق ذلك بالمرصف R13 من PROGk

(3) يتعلّق ذلك بالمرصف R13 من PROGj .

```
  LOC  OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT

000000                                 1 PROGJ    CSECT
000000                                 2 PROLOGUE DS     0H
                                       3 *        SAUVEGARDE DES REGISTRES DE L'APPELANT
000000 90EC D00C      0000C            4          STM    14,12,12(13)
                                       5 *        DEFINITION ET CHARGEMENT DU REGISTRE DE BASE
                            00000      6          USING PROGJ,12             LE REG 12 EST PRIS POUR BASE
000004 18CF                            7          LR     12,15               ADRESSE PROGJ DANS 12
                                       8 *        SAUVEGARDE DE R13 DANS LA SAVE AREA DE CE PROGRAMME.
000006 50D0 C01C      0001C            9          ST     13,SAVEAREA+4
                                      10 *        SAUVEGARDE DE L'ADRESSE DE LA SA DE CE PROGRAMME DANS LA SA
                                      11 *                                   DE L'APPELANT
00000A 182D                           12          LR     2,13
00000C 41D0 C018      00018           13          LA     13,SAVEAREA
000010 5002 0008      00008           14          ST     13,8(2)
000014 47F0 C060      00060           15          B      DEBUT
                                      16 *        DEFINITION DE LA SAVE AREA
000018                                17 SAVEAREA DS     18F
000060                                18 DEBUT    DS     0H
                                      19 *        ------
                                      20 *        ------
                                      21 *        ------
                                      22 *        SEQUENCE D'APPEL DE FROGK
000060 58F0 C070      00070           23          L      15,=V(PROGK)
000064 05EF                           24          BALR   14,15
                                      25 *        ------
                                      26 *        ------
                                      27 *        ------
000066                                28 EPILOGUE DS     0H
                                      29 *        SEQUENCE DE RETOUR VERS PROGI
000066 58D0 C01C      0001C           30          L      13,SAVEAREA+4
00006A 98EC D00C      0000C           31          LM     14,12,12(13)
00006E 07FE                           32          BR     14
                                      33          END
000070 00000000                       34                 =V(PROGK)
```

EXTERNAL SYMBOL DICTIONARY

SYMBOL	TYPE	ID	ADDR	LENGTH	LDID
PROGJ	SD	0001	000000	000074	
PROGK	ER	0002			

التعليمة STM تسمح بترتيب مراصف متتالية عند كل رغبة باستعمال مراصف متجاورة .

إتفاقات الربط المعرّفة سابقاً تسمح بطلبات المناداة الداخلة ضمن البرامج . وهي لا تسمح أبداً بإجزاء طلبات مناداة تكرارية تحتاج إلى تعريف مكدس (STACK) خزن للنص . هذه الأواليات ليست موضوع هذا الكتاب . ولكن نشير إلى أن النظام OS يضع بتصرُّف المستعمل الوسائط لتعريف وإدارة منطقة من الذاكرة لكتابة برامج تكرارية (ماكرو GETMAIN) .

وللحاجة إلى التناسق والتوافق ، فإن المبرمج سيقوم بنفس عمليات الإختيار كالنظام OS في استعمال المراصف لإجراء الوصلات بين البرامج الثانوية .

22 . التأويل المشروط وماكرو التعليمات

1.22 . التأويل المشروط

التأويل المشروط هو عبارة عن خطوة جديدة في التطور من لغة المكنة إلى اللغة المتطورة . ويتعلّق ذلك بلغة تسمح بإنشاء وتوليد ، في مرحلة ما قبل التأويل ، نص مستهدف (object text) يمكن معالجته بواسطة المؤول . النص المؤول الناتج يمكن ، حسب القيم الأولية المخصصة لمتحولات التأويل المشروط ، أن يتغيّر من تأويل إلى آخر . بإمكاننا مثلاً ، إدخال ، خلال مرحلة إعداد البرنامج ، متتالية من التعليمات (طباعة وسيطية تسمح بمتابعة أثر (trace) البرنامج)التي، بواسطة تعديل بسيط للقيم الأولية لمتحولات التأويل المشروط ،سيتمّ إلغاؤها عند التأويل النهائي . هذه العملية ، مضافة إلى استعمال الماكرو تعليمات[1] تجعل المؤول قريباً من اللغة المتطورة ، وتسمح للمبرمج بأن يُجهِّز بوسائل كالتعليمات : PERFORM... ، DO ... WHILE التي تُسهِّل البرمجة .

من غير الممكن هنا عرض جميع إمكانيات التأويل المشروط . سنحاول عرض الخطوط العريضة لهذه الطريقة بواسطة أمثلة توضح لنا العملية .

1.1.22 . متحولات وثوابت التأويل المشروط

التأويل المشروط يُعالج رموزاً بقيم قابلة للتغيير : وهي عبارة عن متحولات التأويل . تبدأ أسماؤها بالرمز « & » ، وتحتوي على أكثر من ثمان سمات أبجعددية ، بما فيها « & » . السمة الثانية يجب أن تكون حرفاً . متحولات التأويل هي من ثلاثة أنواع A ، B وC أي حسابية ، منطقية وأبجعددية . يمكنها أن تكون مركزية بداخل ماكرو ـ إجراء والكود ـ المفتوح[2] (Open-code) أو شاملة (كلّية) في جميع ماكرو ـ الاجراءات وفي الكود المفتوح . يجب أن يصرَّح عن جميع متحولات التأويل ، المركزية

---

(1) مُصطلح معرَّف في 2.6

(2) الكود المفتوح :(Open code) قسم من كودالمصدر يكون موجوداً خارج وبعد الماكرو ـ تعريفات

والكلّية ، قبل إستعمالها . ويتمّ التصريح حسب نوع المتحوّلة A ، B أو C :

LCLA ...	LCLB ...	LCLC ...	(مركزية)
GBLA ...	GBLB ...	GBLC ...	(كلّية)

لا يجب أن يُصرَّح عن متغيرات الماكروتعريف (فقرة 1.2.22) . عند التصريح توضع المتحولات A وB في صفر ويجري إعداد المتحولة من النوع C في « سلسلة فارغة من السمات » .

لا يمكن بلوغ متحولة مُصرَّح عنها على أنها مركزية إلا في نفس الماكرو تعريف وفي الكود المفتوح . أما المتحولة المصرَّح عنها « شاملة (كلّية) » فيمكن بلوغها من ماكرو تعريفات أخرى .

يمكن أن تكون متحولة التأويل المشروط عبارة من متحولة مؤشّرة ، وفي هذه الحالة يجب أن يتم التصريح عنها في مستوى LCLx أو GBLx ، كما نُصرِّح عن الجدول في فورتران . هكذا فإن :

LCLA &TAB(20)

تصرِّح عن &TAB كجدول من 20 عنصراً نستطيع بلوغه بواسطة أحد الأشكال التالية :

&TAB (تعبير حسابي)	مثلاً : &TAB(&I+3)
&TAB (متحولة مؤشّرة)	مثلاً : &TAB(&VAR(&I))

التعبير الذي يعطي قيمة المؤشر يجب أن يكون إيجابياً وأن لا يزيد عن حجم الجدول المشار إليه في التصريح .

الثوابت الحسابية عبارة عن أعداد صحيحة بإشارة أو بدون إشارة حيث يجب أن تكون قيمتها بين : (-2^{31} و $2^{31}-1$) .

تأخذ الثوابت المنطقية القيمة 0 أو 1 التي تناسب الغلط والصحيح . الثوابت من النوع سلسلة سمات تحتوي على عدد من 0 إلى 255 سمة محصورة بداخل فواصل عليا ، ويمكن أن تكون مؤشّرة .

أمثلة :

'ABCDEF' (4) ، تعادل 'D' ،

'ABCDEF'' (2,3) تعادل 'BCD' ،

المؤشر الأول يعطي الموقع الأولي للسلسلة الثانوية والثاني يعطي طولها .

2.1.22 . أسماء الأوسمة

منطقة الرمز من أمر تأويل مشروط يمكن أن تحتوي على وسم تأويل مشروط . إنّـه عبارة عن رمز يبدأ بالنقطة « . » ويسمح ببلوغ أمر تأويل مشروط . لأسماء الوسم مدى مركزي .

3.1.22 . أوامر التخصيص SETx

تقوم بتخصيص قيمة معينة إلى متحولة التأويل المشروط ، تتعلق بنوع المتحولات A ، B ، وC وتتم بواسطة SETA ، SETB أو SETC . نشير إلى أن متحولة التأويل التي تحصل على التخصيص موجودة في المنطقة المحجوزة عادة للوسم . ولو إفترضنا ان &A ، &B و&C هي متحولات من النوع A ، B وC . نكتب :

منطقة الرمز	منطقة العملية	منطقة المعامل
&A	SETA	تعبير حسابي
&B	SETB	(تعبير منطقي)
&C	SETC	'تعبير أبجعددي'

وبشكل عام ، يحسب التعبير وتُخزّن القيمة الناتجة في متحولة التأويل الموجودة لجهة اليسار .

التعابير الحسابية

وتُكتب بواسطة المؤثرات + ، − ، * و/ (قسمة صحيحة بدون باق) . التقييم يتم من اليسار إلى اليمين بقواعد الأولوية العادية .

أمثلة :

			القيمة التي تأخذها المتحولة
&A1	SETA	10	10
&A1	SETA	&A1+1	11

التعابير المنطقية

تُكتب بداخل أهلّـة بواسطة المؤثرات NOT ، AND وOR المذكورة في الترتيب التناقصي للأولويات . وبفضل وجود مؤثرات العلاقة بإمكاننا إجراء المقارنات بين التعابير الحسابية .

مؤثرات علاقة :	LT	LE	EQ	NE	GE	GT
المعنى	<	≤	=	≠	≥	>

يجب أن تكون المؤثرات محاطة بفراغات .

أمثلة :

```
&B4     SETB    (&B1 OR &B2 AND &B3)
&B5     SETB    (&A1 GT &A2)
&B6     SETB    ('&C' EQ 'ALLOC')
```

تعابير من نوع سلسلة سمات

هي عبارة عن مجموعات من الثوابت والمتحولات من النوع الابجعددي المحصورة بداخل فواصل عليا . المؤثر « . » (نقطة) يسمح بإجراء عمليات الإتحاد[1] . الترميز المؤشر يسمح باستخراج السلاسل الثانوية .

أمثلة :

```
                                        القيمة التي
                                        تأخذها المتميزة
&C1     SETC    'CHA'                   CHA
&C2     SETC    '&C1'                   CHA
&C3     SETC    '&C1'.'INE'             CHAINE
             ou '&C1.INE'
&C4     SETC    'CHAINE'(2,5)           HAINE
                  الرتبة ↑ ↑ الطول
&C5     SETC    '&C4'(1,3).'&C4'(5,1)   HAIE
&C6     SETC    'L''NOM'                L'NOM
&C7     SETC    '5'                     5 (caractère)
&C8     SETC    '&C7..25'               5.25 (un seul point)
&C9     SETC    '&A+10'                 si &A = 10 alors
             ou '&A.+10'                10+10 et non 20
&C10    SETC    '&C1&C1'                CHACHA
             ou '&C1.&C1'
```

نشير (&C10) إلى أن النقطة في عملية الإتحاد هي إختيارية عندما نجمع بين متحولتين من السمات لأن الفاصل & لا يسمح بقيام أي نوع من الإبهام .

عندما تدخل المتحولات من النوع A إلى يمين الأمر (&C9) SETC ، فإن قيمة المتحولات تستبدل بالمتحولات ولكن بدون إجراء لأية عملية .

التعابير من النوع سلاسل السمات هي مهمة لأنها تسمح بإنشاء رموز أو بناء تعليمات إتحاد متتالية . هناك أمثلة توضح إستعمالها عند دراسة الماكرو ـ إجراءات .

---

(1) عملية الربط ـ جمع سلسلتين ABCD وEF معناه تشكيل السلسلة ABCDEF .

4.1.22 . أوامر التفريع إلى أوسمة التأويل

التفريع الالزامي يتم بواسطة AGO والتفريع المشروط بواسطة AIF . ويُكتبان :

وسم للتأويل المشروط AGO [وسم التأويل المشروط]

وسم تأويل مشروط (تعبير منطقي) AIF [وسم تأويل مشروط]

أمثلة :

```
AGO    .SUITE
AIF    .('&C' EQ 'OUI').ET1
```

إذهب إلى SUITE

إذا &C تعادل OUI (نعم)
إذهب إلى ET1 ، وإلا تابع بالتالي .

5.1.22 . الأمر ANOP

هو أمر « بدون عملية » يسمح بتعريف وسم معين (Label) . ويُستعمل بشكل خاص عندما نرغب بإجراء تفريع إلى أمر (توجيه) SETx ، ويكون حقل الوسم العادي مشغولاً بمتحوّلة .

```
         AGO     .SUITE
         ---     ------
.SUITE   ANOP
&VAR     SETA    &VAR+1
```

6.1.22 . أمثلة على إستعمال التأويل المشروط

سنذكر عدة أمثلة عند دراسة ماكرو ـ الإجراءات . هنا نكتفي بتفصيل بعض النقاط

مثل 1

نرغب ، خلال تنفيذ البرنامج ، بإجراء تأويل مجموعة من التعليمات (طباعة وسيطية مثلاً) بإلغاء تعليمات التأويل النهائية دون سحب البطاقات المناسبة لها . سنُخضِع إذاً تأويل هذه التعليمات للقيمة التي تأخذها متحولة التأويل التي تدعى هنا & TEST

```
&TEST    SETA    1                   (مرحلة البدء بالعمل)
         ----    ---
         AIF     (&TEST EQ 0).SAUT
         ---     ---                 تعليمات للتأويل
         ---     ---                 خلال مدة الاختبار
.SAUT    ---     ---
```

بجعل المتحولة &TEST تعادل صفراً نكون قد ألغينا تأويل هذه التعليمات .

مثل 2

إنشاء نصّ معيّـن .

التأويل المشروط يمكن أن يُستعمل لانشاء نصّ متحوّل من تأويل إلى آخر . يمكن لهذا النصّ أن يكون رمزاً أو تعليمة .

يؤدي إلى توليد الأمر :	R&NO	EQU	&NO
إذا كانت المتحولة &NO تعادل 1	R1	EQU	1

2.22 . الماكرو - إجراءات

باستعمال الماكرو إجراءات تجد أوالية التأويل المشروط فائدتها .

الماكرو إجراء هو عبارة عن برنامج يحمل إسماً مؤلفاً من سلسلة من التعليمات وأوامر التأويل المشروطة وغير المخصورة بالأوامر MACRO وMEND .

مثلاً : الماكرو تعريف التالي :

لائحة المتغيرات الشكلية إسم

```
        MACRO
        SOMME   &U,&V,&W
        L       1,&U
        A       1,&V
        ST      1,&W
        MEND
```

سطر نموذجي

جسم الإجرا

سيكون الماكرو تعريف موجوداً خارج البرنامج (open code) الذي يُراجعه . بإمكان الماكرو تعريف أن يكون موجوداً في مكتبة المُستعمل أو مكتبة المؤول .

الماكرو تعليمات هي إذاً السطر من البرنامج الذي يطلب من المؤول إدخال نص النموذج في البرنامج باستبدال المتغيرات الشكلية بالمتغيرات الفعلية .

مثلاً :

```
SOMME   A,B,C
L       1,A
A       1,B
ST      1,C
```

يولّـد المتالية .

المتغيّـرات الوسيطية الفعلية

نفترض عندئذٍ بأن هذا النظام ، المزود بالتأويل المشروط ، يسمح بإنشاء نماذج ستاندارد لبرامج يقوم المؤول بجعلها متوافقة مع كل حالة خاصة حسب قيم تحولات التأويل المشروط .

1.2.22 . تنقّل المتغيّرات

كما في حالة البرامج الثانوية ، المتغيّرات الشكلية هي متغيّرات السطر النموذجي في الماكرو تعريف والمتغيّرات الفعلية هي متغيّرات الماكرو تعليمة . المتغيّرات الشكلية هي رموز تسبقها السمة « & » .

يتكوّن السطر النموذجي في الماكرو تعريف على الشكل التالي :

اسم الإجراء	لائحة المتغيّرات الشكلية
PROC	&U,&NO=3,&QTE=,&V,&RES=5,&W,&X

قيم نحو النقصان (0 أو سلسلة فارغة إن لم يجرِ تحديدها) .

المتغيّرات الشكلية هي على نوعين :

- متغيّرات الوضع : &U, &V, &W و&X في المثل ،
- متغيّرات الكلمة المفتاح : &NO ، &QTE و&RES . ونميّزها بكون أسمائها متبوعة بالرمز « = » وربّما بالقيمة التي تأخذها نحو النقصان ، قيمة تساوي « السلسلة الفارغة » في حال عدم تحديدها . ويتكوّن سطر نداء الماكرو تعريف كما يلي :

اسم الإجراء	لائحة المتغيّرات الفعلية
PROC	RES = 6, A, B, QTE = 4,,D

1 ـ متغيّرات مرتبطة بالمتغيّرات الشكلية ـ من حيث مواقعها في اللائحة . لدينا هكذا التناسب بين A و&U ، B و&V ، D و&X . إنّ فاصلتين متتاليتين تشيران إلى غياب متغيّر الوضع .

2 ـ متغيّرات الكلمة المفتاح : الوصل بين المتغيّرات الشكلية والفعلية القائم بفضل تشابه الاسم . هذه العناصر يجب أن يليها الرمز « = » وربّما قيمة تعدّل القيمة المحدّدة نحو النقصان . في مثلنا تأخذ RES القيمة 6 ، QTE القيمة 4 وتحتفظ NO بالقيمة 3 نحو النقصان .

3 ـ قد تكون لوائح متغيّرات محاطة بأهلّة . لنأخذ الماكرو تعليمة :

```
PROC 1      (A, B, C, D), K = (E, F, G, H)
```

والسطر النموذجي المناسب :

```
PROC 1      &POS,&K=
```

تتكوّن المتغيّـرات الفعلية بواسطة اللائحتين (A, B, C, D) و(E, F, G, H) . أمّـا &POS(3) فيُستبدَل عندئذٍ بـ C خلال انتشار الماكرو تعليمة . كذلك يُستبدَل &K(2) بـ F . بإمكان لوائح المتغيّـرات أن تكون ذات أطوال متغيّـرة ، وسنرى أنّ الخاصيّة N' & POS تسمح بمعرفة طول اللائحة المرتبطة بـ POS .

2.2.22 . تطبيق

المثل التالي يقوم بتوليد تعليمات تسمح بجمع n خلية من الذاكرة منقولة إلى ماكرو الإجراء بواسطة لائحة &RES. ستحتوي على النتيجة و&NB تُمثـل عدد العناصر المطلوب جمعها . المؤشر المركزى &I يُستعمل لمراجعة مختلف عناصر اللائحة .

```
           MACRO
           SOMME &MEM,&RES,&NB=,&REG=
           LCLA  &I
           L     &REG,&MEM(1)
&I         SETA  1
.BOUCLE    ANOP
&I         SETA  &I+1
           AIF   (&I GT &NB).FIN
           A     &REG,&MEM(&I)
           AGO   .BOUCLE
.FIN       ST    &REG,&RES
           MEND
```

```
000060 5830 C074   00074   64         SOMME (A,B,C,D),X,NB=4,REG=3
000064 5A30 C078   00078   65+        L     3,A
000068 5A30 C07C   0007C   66+        A     3,B
00006C 5A30 C080   00080   67+        A     3,C
000070 5030 C084   00084   68+        A     3,D
                           69+        ST    3,X

000074                     72 A       DS    F
000078                     73 B       DS    F
00007C                     74 C       DS    F
000080                     75 D       DS    F
000084                     76 X       DS    F
```

3.2.22 . الأمر MEXIT

يسمح بوقف تأويل الماكرو تعريف . من الممكن إعتباره معادلاً للتفريع إلى الأمر MEND .

4.2.22 . الأمر ACTR

يسمح بمراقبة عدد AIF وAGO الجاري خلال التأويل المشروط . ويكتب :

(تعبير حسابي) ACTR

يؤدي إلى توليد عداد يعادل مضمونه قيمة التعبير الحسابي . يمكن أن يكون العداد مركزياً للماكرو تعريف أو شاملاً . في كل مرّة يجري فيها تنفيذ AIF أو AGO بواسطة المؤول ، فإن العـداد المناسب لهـذا القسم من البـرنـامـج يُخـفّـض واحـداً مـن مضمونه . وعندما يبلغ الصفر ، فإن المؤول يخرج من الماكرو تعريف (فعل مَعادل لِـ

MEXIT) أو يُوقف التأويل إذا كان ذلك متعلقاً بعداد شامل . هذا الأمر يسمح بتحديد عدد الحلقات التي تجري في مرحلة ما قبل التأويل .

5.2.22 . الأمر MNOTE

يمكن أن يُستعمل من قِبل المبرمج لتوليد رسالة الخطأ الخاصة به أو طباعة قيم وسيطية مأخوذة من متحولات التأويل .

ويمكن أن يُكتب بعدة أشكال :

وسم تأويل

```
(1)  étiquette        MNOTE   code,'message
     d'assemblage
(2)  étiq. assem.     MNOTE   ,'message'
(3)  étiq. assem.     MNOTE   *,'message'
(4)  étiq. assem.     MNOTE   'message'
```

الكود هو عبارة عن تعبير حسابي بقيمة محصورة بين 0 و255 يربط مستوى من الخطأ بالرسالة . في الشكل 2 يُفترض بالكود أن يكون مُعادلاً لـ 1 . لا تُطبع الرسالة من ضمن رسائل الخطأ إلا إذا كان الكود الذي يشير إلى درجة الحقيقة هو أعلى من أو يعادل الكود المعتمد من المؤول .

الشكلان 3 و4 يولّدان الرسالة كمجرّد ملاحظية . .

6.2.22 . الملاحظيات :

من الممكن إدخال ملاحظيات في ماكرو التعريفات على الشكل التالي :

```
* COMMENTAIRE GENERE
.* COMMENTAIRE NON GENERE
```

7.2.22 . الدوال من النوع الذاتي (Intrinsic)

&SYSLIST

تسمح ، داخل الماكرو تعريف ، بتسمية متغيرات الموقع الموجودة داخل ماكرو تعليمة النداء . وتُكتب بمؤشر أو بمؤشرين يمكن أن يكونا عبارة عن تعابير حسابية من نوع ذلك الذي رأيناه في الفقرة 3.1.22 . سنختبر إستعمالها بالخاصية 'N .

&SYSLIST(&I) تشير إلى المتغيِّر الفعلي الخاص بالموقع رقم i من التعليمة . يمكن أن يكون هذا المتغير الفعلي عبارة عن لائحة (حسب الفقرة 3.1.22). في هذه الحالة ، سنسمّي العنصر رقم j من اللائحة بالرتبة &I بواسطة &SYSLIST (&I,&J) . في المثل المذكور في الفقرة 2.2.22 &SYSLIST(1,2) تعني المتغيِّر B ، و&SYSLIST(2) تعني X .

&SYSLIST(0) تعني الوسم الموجود قبل الماكرو تعليمة الخاصة بالنداء . هذه المهمة تسمح بتفادي تسمية المتغيرات .

&SYSNDX

هي عبارة عن عداد من أربعة أرقام عشرية ، وهو مركزي ضمن ماكرو ـ تعريف ، وتزداد قيمته عند كل استعمال جديد للماكرو . لا يمكن أن يُستعمل وحيداً ولكن يُمكن أن يتّحد مع رمز ما . هذه هي الوسيلة لتوليد أوسمة مختلفة عند كل نداء لماكرو ـ التعريف وتسمح بتفادي الأخطاء في التأويل والناتجة عن تعريف الرموز .

مثلاً :

لنفترض الماكرو ـ تعريف التالي :

```
            MACRO
            PROC    &A,...
&A&SYSNDX   ----
            ----
R&SYSNDX    ----
            ----
            MEND
```

النداء الأول يتم بواسطة PROC ETIQ,...

المتحولة &A&SYSNDX تأخذ القيمة ETIQ0001

المتحولة R&SYSNDX تأخذ القيمة R0001

في النداء الثاني بواسطة . . . PROC ETIQ,...

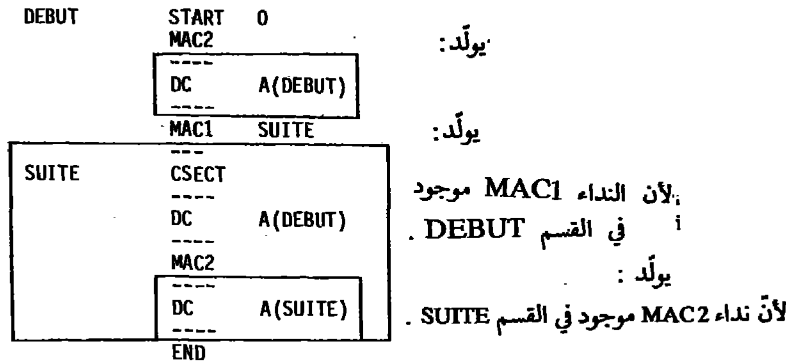
المتحولة &A&SYSNDX تأخذ القيمة ETIQ0002

المتحولة R&SYSNDX تأخذ القيمة R0002

&SYSECT

تسمح بتعريف اسم القسم حيث توجد الماكرو ـ تعليمة المُنادية . المثل التالي يوضح ذلك :

```
        MACRO
        MAC1    &ETIQ
        ----    -----
&ETIQ   CSECT
        ----
        DC      A(&SYSECT)
        ----
        MAC2
        ----
        MEND
        MACRO
        MAC2
        ----
        DC      A(&SYSECT)
        ----
        MEND
```



&SYSPARM

يعطي وسيلة الرجوع إلى المتغيّر SYSPARM لبطاقة EXEC// في Job JCL) Control Language : لغة مراقبة العمل) .

مثلاً :

```
//        EXEC ASMC,PARM=SYSPARM(DEBUG)
//ASM.SYSIN DD *
TEST      START   0
          ----
          AIF     ('&SYSPARM' NE 'DEBUG'). (قفزة)
          ----    ولادة تعليمات
          ----    تنفيذ وتقويم
.SAUT     ANOP
```

&SYSTIME

يعطي ساعة التأويل بواسطة خمس سمات : h.h.mm

&SYSDATE

يعطي التاريخ بواسطة ثمان سمات : mm/jj/aa

8.2.22 . الخاصيات

مفهوم الخاصية المرتبط بمعطى أو بتعليمة جرت إثارته في الفقرة 2.3.6 .. كما

إستعملنا الخاصية ـ طول (فقرة 3.2.7) . يسمح المؤول لنا باستعمال خاصيات أخرى حيث البعض منها يجد إستعمالاً بسبب وجود إمكانيات التأويل المشروط .

الخاصية : 'TYPE T

وقيمتها سمة أبجدية حسب نوع الرمز المُطبَّقة عليه . إذا كانت NUM ، مثلاً ، عبارة عن ثابتة عشرية موسعة ، فإن قيمة T'NUM ستكون Z . الحرف الذي يميِّز النوع هو نفسه المُستعمل في الأوامر A:DC تناسب ثابتة عنوان من نوع A ، بينما B تناسب ثابتة منطقية . . . ونضيف التناسبات التالية :

G	ثابتة بفاصلة ثابتة وطول محدَّد ظاهر
K	ثابتة بفاصلة متحركة وطول محدد ظاهر
R	ثابتة عنوان بطول محدَّد ظاهر
I	تعليمة ـ آلية
M	ماكرو تعليمة
W	CCW
J	اسم قسم
T	رمز خارجي
N	قيمة تعريف أوتوماتيكي } تتعلَّقان بمتغيرات الماكرو تعليمة
O	سمة محذوفة

الخاصية 'LONGUEUR L (طول)

جرت دراستها في الفقرة 3.2.7 .

الخاصية مقياس 'S

عبارة عن قيمة رقمية تتعلَّق بنوع الرمز .

ـ لعدد عشري (نوع P أو Z)

عبارة عن عدد الأرقام في القسم الكسري .

ـ لعدد بفاصلة متحركة (أنواع L, E, D أو K)

إنَّـه عدد الأصفار السادس عشرية في يسار القسم العشري (الوزن الأكبر) .

ـ لعدد بفاصلة ثابتة (الأنواع M ، F أو G)

عبارة عن القوة 2 التي يتم ضرب قيمة الثابتة بها . وتشير إلى عدد البتات في القسم الكسري إذا كان إيجابياً ، وعدد البتات المتروكة إذا كان سلبياً .

الخاصية قسم صحيح I'

عبارة عن عدد يتعلّق بـ S' و L' .

- لعدد عشري من نوع P $I' = 2 * L' - S' - 1$
- لعدد عشري من نوع Z $I' = L' - S'$
- لعدد بفاصلة متحركة من نوع K ، L ، E ، D $I' = 2 * (L' - 1) - S$ مع $L' \leq 8$
- لعدد بفاصلة متحركة من نوع L $I' = 2 * (L' - 1) - S' - 2$ مع $L' > 8$
- لعدد بفاصلة ثابتة من نوع G ، F ، H $I' = 8 * L' - S' - 1$

الخاصية عدد السمات K'

وتُطبَّـق فقط على مُتغيراتِ الماكرو ـ تعليمة وأيضاً ، بإشراف OS ، على الرموز المتحولة . . & وعلى الدوال الذاتية (من نوع intrinsic) . وتعطي عدد سمات الرمز التي تطبَّـق عليه .

أمثلة : في مثل الفقرة 2.2.22 : K'&MEM = 9

```
&A SETA 253 : K'&A = 3,
&B SETB 0   : K'&B = 1,
&C SETC 'ALPHA' : K'&C = 5.
```

الخاصية عدد العناصر من اللائحة N'

وتنطبق فقط على متغيرات الماكرو ـ تعليمة ، وتعطي عدد عناصر اللائحة .

مثلاً :

خط نموذج	PROC &A,&B,&K=
ماكرو تعليمة	PROC (1,2,,4),U,K=3
(يتم تعداد السمات غير الموجودة)	N'&A = 4
مُتغيرات الموقع	N'&SYSLIST = 2
	N'&SYSLIST(1) = 4.

9.2.22 . أمثلة عن الماكرو ـ تعريفات

الماكرو ـ تعريف التالي يسمح بتوليد الأوامر (التوجيهات) المُعادلة لـ RiEQUI

```
         MACRO
         EQUREG
*        MACRO D'EQUIVALENCE REGISTRES     ماكرو معادل المراصف
         GBLA  &NO
&NO      SETA  1
.SI      AIF   (&NO GT 15).FIN
R&NO     EQU   &NO
&NO      SETA  &NO+1
         AGO   .SI
.FIN     MEND
```

ويُولِّـد الكود التالي :

```
         33          EQUREG
         34+*        MACRO D     ماكرو معادل المراصف
00001    35+R1       EQU     1
00002    36+R2       EQU     2
00003    37+R3       EQU     3
00004    38+R4       EQU     4
00005    39+R5       EQU     5
00006    40+R6       EQU     6
00007    41+R7       EQU     7
00008    42+R8       EQU     8
00009    43+R9       EQU     9
0000A    44+R10      EQU     10
0000B    45+R11      EQU     11
0000C    46+R12      EQU     12
0000D    47+R13      EQU     13
0000E    48+R14      EQU     14
0000F    49+R15      EQU     15
```

الماكرو ـ تعريف PROLOGUE يسمح بشحن واحد أو عدة مراصف قاعدة مخزَّناً مفهوم البرنامج المُنادي حسب المعايير العادية المُحدَّدة في الفصل 21 . وهو يُعرِّف في نفس الوقت منطقة SAVE AREA للبرنامج الجاري . عنوان القاعدة الذي جرى اختياره هو عنوان نقطة الدخول إلى البرنامج . ويرد الكود المولَّـد على الصفحة التالية .

```
              MACRO
              PROLOGUE  &ABASE=,&RBASE=
   PROLOG     EQU    *
              LCLA   &I,&D
              LCLC   &CH
              STM    14,12,12(13)
   .*         CONSTRUCTION DE USING     إنشاء مصنع
   &CH        SETC   '&RBASE(1)'
   &I         SETA   2
   .T1        AIF    (&I GT N'&RBASE).SUIT1
   &CH        SETC   '&CH'.','.'&RBASE(&I)'
   &I         SETA   &I+1
              AGO    .T1
   .SUIT1     ANOP
              USING  &ABASE,&CH
              LR     &RBASE(1),15
              ST     13,SAVEAREA+4
              LR     2,13
              LA     13,SAVEAREA
              ST     13,8(2)
   .*         CONSTRUCTION DES CHARGEMENTS DES REGISTRES DE BASE
   &I         SETA   2
   .T2        AIF    (&I GT N'&RBASE).SUIT2      إنشاء شحن المراصف القاعدي
              L      0,=F'4096'
   .T3        AIF    (&I GT N'&RBASE).SUIT2
              AR     15,0
              LR     &RBASE(&I),15
   &I         SETA   &I+1
              AGO    .T3
   .SUIT2     ANOP
              B      *+76
   SAVEAREA   DS     18F
              MEND
```

من المفيد دراسة أمثلة الماكرو تعريفات المذكورة في كتاب إ. تابورييه Y.Tabourier ، أ. روشفلد Rochfeld ونس. فرانك C. Frank . إنّـها عبارة عن ماكرو تعريفات تسمح ببناء برنامج مؤوّل بصورة بنيوية مركّبة . والكتاب يعرض للماكرو WHILE زائد شرط ، DO ، ENDWHILE ، IF ، THEN ، ELSE ، ENDIF ، BLOCK وENDBLOCK بالتفصيل ، وتقوم هي باستدعاء 2 ماكرو تديران مكدساً من المؤثرات .

```
                                         52            PROLOGUE ABASE=P1,RBASE=(12,11)
                                 00000   53+PROLOG     EQU    *
000000 90EC D00C          0000C          54+           STM    14,12,12(13)
                                 00000   55+           USING  P1,12,11
000004 18CF                              56+           LR     12,15
000006 50D0 C024          00024          57+           ST     13,SAVEAREA+4
00000A 182D                              58+           LR     2,13
00000C 41D0 C020          00020          59+           LA     13,SAVEAREA
000010 50D2 0008          00008          60+           ST     13,8(2)
000014 5800 C090          00090          61+           L      0,=F'4096'
000018 1AF0                              62+           AR     15,0
00001A 18BF                              63+           LR     11,15
00001C 47F0 C068          00068          64+           B      *+76
000020                                   65+SAVEAREA   DS     18F
```

```
                                         52            PROLOGUE ABASE=P1,RBASE=12
                                 00000   53+PROLOG     EQU    *
000000 90EC D00C          0000C          54+           STM    14,12,12(13)
                                 00000   55+           USING  P1,12
000004 18CF                              56+           LR     12,15
000006 50D0 C01C          0001C          57+           ST     13,SAVEAREA+4
00000A 182D                              58+           LR     2,13
00000C 41D0 C018          00018          59+           LA     13,SAVEAREA
000010 50D2 0008          00008          60+           ST     13,8(2)
000014 47F0 C060          00060          61+           B      *+76
000018                                   62+SAVEAREA   DS     18F
```

23 . نصائح في البرمجة

ليس هدفنا عرض طريقة في البرمجة تشبه البرمجة الإنشائية ، ولكن ببساطة إعطاء بعض النصائح الناتجة عن الخبرة العملية لمختلف الطرق . هذه الملاحظات يمكن أن توسّـع لتشمل جميع أنواع المؤوّل وفي بعض الأحيان تنطبق على اللغات المتطورة .

1.23 . تركيبة المُعالجة

1.1.23 . البرمجة الزجلية

هي عبارة عن قاعدة عامة في البرمجة . هناك فائدة من تقسيم المسألة إلى زُجل (أقسام) صغيرة قدر الإمكان . كلّ زجلة تحلّ مهمة معينة والبرنامج الرئيسي يؤمن ترابط الأقسام فيما بينها . ولقد عرضنا في الفصلين 20 و21 . طريقة استعمال وسائل التقطيع وإنشاء البرامج - الثانوية .

2.1.23 . تقديم وإعداد

البرنامج بلغة المؤوّل هو عادة عبارة عن نص غير واضح ، ويجد المُصمِّم صعوبة في تعديل وإعادة قراءة ما كتبه منذ اللحظة التي يترك فيها برنامجه جانباً لبعض الوقت . يجب إذاً كتابة الملاحظات بعد كل تعليمة لتوضيح نصّ البرنامج . الأوامر SPACE N (إدخال عدة أسطر n بيضاء) ، EJECT (عبور إلى الصفحة التالية) وPRINT NOGEN (إلغاء توليد كود الماكرو تعليمات) تسمح بتسهيل نصّ البرنامج بجعله أكثر وضوحاً .

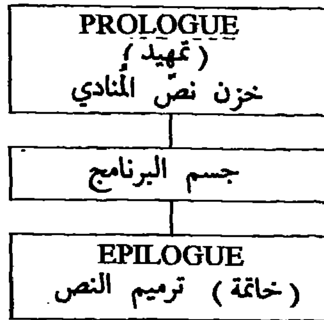
البرنامج المُزوَّد بملاحظيات يبدأ بتحديد مهمة الزجلة ، وروابطها مع الزجل الباقية كما يحتوي على أسماء ومهمة المتحولات والمراصف المستعملة .

2.23 . تركيبة الزجلة

1.2.23 . التمهيد والخاتمة (Prologue، epilogue)

بإمكاننا إعتبار كل برنامج وكأنه برنامج ثانوي لبرنامج آخر . الزجلة الرئيسية هي عبارة عن برنامج ثانوي من نظام التشغيل ويجب عليها أن تخزّن نتائج البرنامج المُنادي .

تقنية الخزن وترميم نصّ المُنادي هي أساسية وقد جرى تعريفها في الفصل 21 . بإمكاننا إنشاء كل زجلة على الشكل التالي :



هناك فائدة للمبرمج في تحقيق التمهيدات والخاتمة الخاصة به حسب القواعد المُتفق عليها والمذكورة في الفصل 21 . الاتصال بين الزجل المكتوبة في اللغات المختلفة سيكون مبسّطاً وأكثر من خطأ سيتم تفاديه باستعمال مناسب للمراصف . لقد ذكرنا مثلاً في الفصل 22 الماكرو ـ تعليمة PROLOGUE التي تحلّ هذه المسألة وتُوفّر على المبرمج كتابة صعبة للتعليمات الأولية .

2.2.23 . جسم البرنامج

يتألف من تعليمات قابلة للتنفيذ ومن معطيات . سنضع المعطيات بعد التعليمات . إستعمال الأمر LTORG سيسمح لنا بوضع تأويل الثوابت من نوع حرفي في المكان الذي نرغب فيه . المتحولات والثوابت ستكون إذاً متراصّة ، مما يجعلها متجاورة في كل dump وستسمح بإجراء تقسيم سهل إلى أقسام إذا كنا نرغب بجعل البرنامج مُبسّطاً للتعديل والاختبار . سنستعمل عند الحاجة أوامر حجز مكان من الذاكرة بواسطة DC أكثر من بواسطة DS معدّين بهذه الطريقة منطقة من الذاكرة بقيمة سوف يمكننا مراقبتها في dump (دلق) .

إستعمال المرجعيات الرمزية

إنّ كتابة LR1,2 تعود عملياً إلى العمل بلغة الآلة . وفي المقابل فإن كتابة LR R1,R2 بعد تعزيف الرمزين R1 وR2 بواسطة EQU معناها إستعمال إمكانيات ومرونة الترميز ، والمرجعان R1 وR2 يظهران في جدول الرموز . من الأفضل أيضاً إعطاء المراصف والمتحولات أسماء مكوّدة حرفياً كما جرى في أمثلة الفصل 15 . فليس من المزعج أكثر من قراءة التعليمات التي تذكر المراصف بشكل ظاهر .

هكذا ، فكتابة 14 + * B تؤدي إلى سيئة تكمن في تجميد البرنامج ، ويصبح من

غير الممكن إدخال تعليمات جديدة بين العنوانين * و14+ * دون تعديل تعليمات التفريع . لذا فمن الأفضل تعريف وسم ALPHA وكتابة B ALPHA . الكتابات من النوع n + * لا يجب أن تستعمل إلا داخل الماكرو ـ تعريفات . وختاماً يجب على البرنامج أن يكون دائماً مكتوباً مع أخذ التعديلات اللاحقة بعين الإعتبار إضافة إلى مسائل الصيانة .

هكذا يجب تعريف جميع العناصر القابلة للتعديل في البرنامج بواسطة EQU . هذا الأمر هو شديد الأهمية . وفي حالة التعديل فهو يسمح بتخفيض عدد التغيرات المطلوب إجراؤها . ويقدم فائدة تكمن في جعل التعليمات « مزودة بملاحظيات » . إنّ التمرين 8.13 يوضح لنا ذلك .

الخاصية ـ طول

تسمح بجعل البرنامج يحتوي على متغيّرات . كل تعديل على طول المنطقة لن يؤثّر على التعليمات التي تذكر هذا الطول بواسطة L'ZONE .

تركيبة منطقة المعطيات

بدلاً من مراجعة أقسام (field) نفس المنطقة بواسطة المسافة بالنسبة لبداية المنطقة ، من الأفضل تخصيص (بواسطة EQU) أسماء رمزية لمختلف هذه الأقسام . كل تعديل على التركيبة يصبح عندئذٍ سهلاً . يُوضح لنا التمرين 2.8 تعريف تركيبة كهذه .

إستعمال الكود الحرفي

يترك للمؤول مهمة تعريف الثوابت الضرورية دون إسهاب . هذه الثوابت يمكن أن تكون مجموعة في المكان المطلوب بواسطة الأمر LTORG .

كتابة الأوسمة

سنعرِّف الأوسمة بواسطة الأمر DSOH . نتأكّد من تسطير (إصطفاف) حدود نصف ـ كلمة والوسم لن يعود مرتبطاً بالتعليمة الموجودة في الجهة المقابلة له . سيصبح ممكناً عكس بعض التعليمات بواسطة مُعالجة بسيطة للبطاقات .

إستعمال المراصف

قبل أية عملية برمجة يجب التنقيب عن الخيارات التي يقوم بها النظام لاستعمال المراصف . وقد جرى عرض ذلك في الفصل 21 . وللمبرمج فائدة في إجراء نفس الاختيار لأسباب تتعلّق بالتوافق . فلنذكّر أن OS :

يشحن في R15 عنوان نقطة الدّخول ،

في R14 عنوان العودة ،

في R13 عنوان المنطقة SAVE AREA .

ويستعمل R0 لارسال نتيجة مهمّة من نوع (FUNCTION في فورتران) ،

و R1 لإرسال عنوان لائحة متغيرات إلى برنامج ثانوي .

بعض التعليمات (TRT, EMDK) تستعمل المرصفين R1 وR2 . سيختار المبرمج مراصف القاعدة من ضمن المراصف 12 ، 11 و. . . ومراصف العمل من ضمن المراصف 3 ، 4 ، . . .

إستعمال الماكرو ـ لغة (MACRO-language)

باستعمال الماكرو لغة فإن المؤول يقترب من اللغة المتطورة . وهي تسمح للمبرمج بأن يكون مزوّداً بوسائل إعداد البرنامج وجعله إنشائياً (مركّباً) . وسيكون بإمكانه ، مثلاً ، إنشاء ماكرو ـ تعريف يسمح له بمتابعة أثر البرنامج عند التنفيذ بواسطة طباعة الأوسمة خلال مرحلة الاختبار . عند التأويل النهائي فإن توليد الماكرو ـ تعليمة سيتم إلغاؤه بواسطة تعديل بسيط لقيمة متحولة التأويل . ولن تولّد أوسمة بواسطة ETIQ DS OH . بإمكان المبرمج أن يقوم أيضاً بإنشاء ماكروتعريفات تولّد مثلاً تعليمات من نوع WHILE ، DO ، ENDDO . . . وبإمكان الأوسمة أن تختفي من النص المطلوب تأويله ويُصبح البرنامج أكثر إنشائياً .

وفي النهاية فإنّ الزجلة يمكن أن تحصل على التركيبة التالية :

ماكرو تعليمات	MACRO-DEFINITIONS
ملاحظيات	COMMENTAIRES
EQU...	EQU ...
مقدّمة	PROLOGUE
خاتمة	CORPS
جسم البرنامج	EPILOGUE
منطقة المعطيات	ZONE DE DONNÉES

3.23 . الخلاصة

بشكل عام لا نؤيّد المبالغة في استعمال الحيل والحذق من قبل المبرمج . فالبرنامج « المتحايل » هو غامض على العموم بالنسبة للقارىء المبتدىء ، وأحياناً تقترب الحيل من الإضمار المبهم ويمكننا هنا تصوّر المشاكل التي قد تعترض عمل فريق صيانة البرامج .

في لغة المؤوّل تختلف المسألة نوعاً ما . فبالإمكان إقامة عدد معيّن من الحيل ضمن نطاق تقنيات الحلّ وفي هذا الإطار يتعيّن على المبرمج أن يعرفها . لقد ذكرنا خلال الأمثلة والتمارين عدداً كبيراً من الوصفات المنتشرة كفاية بشكل يسمح لنا باعتبارها كأدوات أساسية . هذا هو السبب الذي يجعلنا نصرّ على دراستها من قبل القارىء بعناية واهتمام .

حلول التمارين

تمرين 1.2 ـ

النظام 16	النظام 2	النظام 10
F	1111	15
23	10 0011	35
100	1 0000 0000	256
400	100 0000 0000	1024
15C.8	1 0101 1100.1	348.5

تمرين 2.2 ـ

النظام 2	النظام 10	النظام 16
11 1010	58	3A
1111 1111 1111	4095	FFF (=1000-1)
1 1010 0011 1011	6715	1A3B
1010 1011 1100	2748	ABC

تمرين 3.2 ـ المكمّل إلى FFFF : E5C4

المكمّل إلى 2:E5C5

الطرح بواسطة جمع المكمّل إلى 2 (نتحقّق ما إذا كان يحقّ لنا تجاهل المرحّل) النتيجة : 1081 .

1A3B على 32 بتة : 3B 1A 00 00

E5C5 على 32 بتة : C5 E5 FF FF

تمرين 4.2 ـ تكويد الإشارة والقيمة المطلقة : $-4.16^7 + 16^6 + 15.16^5$

التكويد بالمكمّل إلى 2 : $-3.16^7 + 14.16^6 + 16^5$

التكويد بالفاصلة المتحرّكة : $-16^1 (15.16^{-1})$

العكس (الضدّ) بالإشارة والقيمة المطلقة : 41 F0.00 00

العكس بالمكمّل إلى 2 : 3E 10 00 00

العكس بالفاصلة المتحرّكة : 41 F0 00 00 (معاير)

لا يمكن لهذا التمثيل أن يكون تمثيل عدد مكوّد بالنظام DCB (عشري مكوّد ثنائياً) .

تمرين 5.2 -

$$C5\ 03\ 20\ 00 = -16^{5}(3.16^{-2}+2.16^{-3})$$
$$= -\frac{16^{5}}{16}.16(3.16^{-2}+2.16^{-3})$$
$$= -16^{4}(3.16^{-1}+2.16^{-2}) = C4\ 32\ 00\ 00$$

تمرين 1.8 -

```
TAB      DC     100AL1(*-TAB+1)
TAB      DC     100A((*-TAB)/4+1)
```

تمرين 2.8 -

```
NOSS     DS     0CL13                 L'NOSS = 13
SEXE     DS     CL1                   L'SEXE =  1
DATE     DS     0CL4                  L'DATE =  4
ANNEE           DS     CL2            L'ANNEE = 2
MOIS            DS     CL2            L'MOIS  = 2
LIEUNAI  DS     0CL5                  L'LIEUNAI = 5
DEPART          DS     CL2            L'DEPART  = 2
COM             DS     CL3            L'COM    = 3
NO       DS     CL3                   L'NO     = 3
```

تمرين 3.8 - • تأطير على حدّ كلمة

```
         DS     0F
Z1       DS     0CL12
PRIX     DS        ZL8
QTE      DS        ZL4
         ORG    Z1
Z2       DS     0CL14
NO       DS     F
TEXTE    DS     CL10
```

تمرين 1.9 -

```
  LOC   OBJECT CODE     ADDR1 ADDR2  STMT    SOURCE STATEMENT

000000                                  1             CSECT
                              00000     2             USING *,12
000000 5810 C02C           0002C        3             L     B,D
000004 5833 C02C           0002C        4             L     3,D(3)
000008 0000                             5             LR    A,D
         *** ERROR ***
00000A 0000 0000           00004        6             ST    D,X'4'(3,C)
         *** ERROR ***
00000E 5803 000B           0000B        7             L     A,B'1011'(3)
000012 0000 0000           00040        8             L     D,E(B)
         *** ERROR ***
000016 5801 C040           00040        9             L     A,E(B)
00001A D200 A000 C02C 00000 0002C      10             MVC   A(B,C),D
000020 D203 C040 C02C 00040 0002C      11             MVC   E(L'D),D
000026 5820 C030           00030       12             L     2,D+L'D
                              00000    13 A           EQU   0
                              00001    14 B           EQU   1
                              0000A    15 C           EQU   10
00002C                                 16 D           DS    5F
000040                                 17 E           DS    12F
                                       18             END


  ASSEMBLER DIAGNOSTICS AND STATISTICS

 STMT  ERROR CODE   MESSAGE

   5   IFO217       RELOCATABILITY ERROR NEAR OPERAND COLUMN 4
   6   IFO217       RELOCATABILITY ERROR NEAR OPERAND COLUMN 2
   8   IFO217       RELOCATABILITY ERROR NEAR OPERAND COLUMN 2

NUMBER OF STATEMENTS FLAGGED IN THIS ASSEMBLY =       3
HIGHEST SEVERITY WAS    12
```

تمرين 1.11 ـ

```
LA      R,0
SR      R,R
```

حلول أخرى بواسطة « أو المقتصرة » أو « الإزاحات » .

تمرين 2.11 ـ

```
LCR     R,R
```

تمرين 3.11 ـ

```
LH      R,=H'-1'
```

نستعمل كون نصف الكلمة موسّعاً إلى كلمة قبل العملية بواسطة انتشار بتة ذات وزن قوي .

تمرين 4.11 ـ

```
LA      R,2048
LA      R,4095      أو : { LA R,4095
L       R,=F'4096'         { LA R,1(0,R)
```

تمرين 5.11 ـ

```
LA      R,4(0,R)
```

تمرين 6.11 ـ

```
MVI     ZONE,C'*'
MVC     ZONE+1(L'ZONE-1),ZONE
```

نستعمل كون الحركة تتمّ باتية بعد باتية من اليسار إلى اليمين .

تمرين 1.12 ـ

```
N       DC      ...        . عدد التكرارات
R1      EQU     3
        ---     ---
        L       R1,N
TRAIT   ---
        ---                . معالجة التكرار
        ---
        BCT     R1,TRAIT
```

تمرين 2.12 ـ

تسمح الماكرو تعليمة SNAP بالحصول على عمليات دلق («dumps») جزئية في الذاكرة . ويجب أن تسبقها ماكرو OPEN (فتح سجلّ) . في حالتنا الحاضرة يمتدّ الدلق dump من العنوان SNAPDEB حتى العنوان SNAPFIN . تعطي الكلمة PSW عنوان بداية SNAP . وتعطي الجهة

اليمنى من dump ، حتى يكون ذلك ممكناً ، تفسير محتوى الذاكرة الثنائي على شكل سمات . وسيتمرّن القارىء بمحاولة إيجاد محتويات مختلف مناطق البرنامج عبر حساب العناوين من خلال العنوان الأساسي الموجود في المرصف 12 .

(أنظر اللائحة listing في الصفحة اللاحقة) .

تمرين 3.12 ـ

```
  LOC  OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
C000C0
                                       1 DEBUT     START 0
                                       2           PRINT NOGEN,DATA
                                       3 *  عكس سلسلة من السمات
                             00003     4 WORK      EQU   3
                             00004     5 IND1      EQU   4
                             00005     6 IND2      EQU   5
000000                                 7 SNAPDEB   DS    0H
000000                                 8 PROLOGUE  DS    0H
000000 90EC D00C       0C00C           9           STM   14,12,12(13)
                             0000C    10           USING DEBUT,12
000004 18CF                           11           LR    12,15
000006 50D0 C078       0C078          12           ST    13,SAVE+4
00000A 41D0 C074       00074          13           LA    13,SAVE

00000E 4150 C000       00000          15           LA    IND2,0
000012 4140 0005       00005          16           LA    IND1,L'CH1
000016                                17 BCL       DS    0H
000016 4334 C067       00067          18           IC    WORK,CH1-1(IND1)
00001A 4235 C06D       0006D          19           STC   WORK,CH2(IND2)
00001E 4155 00C1       00001          20           LA    IND2,1(IND2)
000022 4640 C016       00016          21           BCT   IND1,BCL
```

تمرين 4.12 ـ

```
BCTR   R,0
```

تمرين 5.12 ـ

```
BALR   R,0
```

تمرين 1.13 ـ

```
XC     ZONE,ZONE       منطقة بطول L
XR     R1,R1           مرصف
NI     OCTET,X'00'     باية
```

تمرين 2.13 ـ لنفترض التعليمة في العنوان INSTR . إذن يوجد كود الطول في INSTR + 1 (تعليمة بنسق SS) .

NI (إعادة تصفير)

OI (حيث XX هو الطول ناقص واحد)

علينا أن نتذكّر أنّه ، بالنسبة للتعليمات من النوع SS ، الطوّل المؤوّل هو الطول ناقص واحد .

تمرين 3.13 ـ

```
XC     ZONE1(L),ZONE2
XC     ZONE2(L),ZONE1
XC     ZONE1(L),ZONE2

XR     R1,R2
XR     R2,R1
XR     R1,R2

PACK   OCTET,OCTET   أو   UNPK   OCTET,OCTET
```

```
000000                                   1 DEBUT    START 0                                     A SOMME (مرصف يحتوي المجموع)
                               00000     2 R0       EQU   0                                             (لا زيادة)
                               00004     3 INDICE   EQU   4                                             (مرجع)
                               00002     4 PAS      EQU   2
                               00003     5 REF      EQU   3
                                         6          PRINT NOGEN,DATA
000000                                   7 SNAPDEB  DS    0H
000000                                   8 PROLOGUE DS    0H
000000 90EC D00C          0000C          9          STM   14,12,12(13)
                               00000    10          USING DEBUT,12                  BASE = REGISTRE 12
000004 18CF                             11          LR    12,15
000006 50D0 C0AC          000AC         12          ST    13,SAVE+4
00000A 182D                             13          LR    2,13
00000C 41D0 C0A8          000A8         14          LA    13,SAVE
000010 50D2 0008          00008         15          ST    13,8(2)
                                        16          OPEN  (POUB,(OUTPUT))

00001E 4140 0000          00000         23          LA    INDICE,0
000022 4120 0004          00004         24          LA    PAS,4
000026 5830 C07C          0007C         25          L     REF,N
00002A 0630                             26          BCTR  REF,0                     N-1 DANS REF
00002C 8930 0002          00002         27          SLL   REF,2
000030 4100 0000          00000         28          LA    R0,0
000034                                  29 BOUCLE   DS    0H
000034 5A04 C080          00080         30          A     R0,TAB(INDICE)
000038 8742 C034          00034         31          BXLE  INDICE,PAS,BOUCLE
                                        32          SNAP  DCB=POUB,PDATA=(REGS,PSW,SA),STORAGE=(SNAPDEB,SNAPFIN)
                                        52          CLOSE (POUB)
000072                                  58 EPILOGUE DS    0H
000072 58D0 C0AC          000AC         59          L     13,SAVE+4
000076 98EC D00C          0000C         60          LM    14,12,12(13)
00007A 07FE                             61          BR    14
00007C 0000000A                         63 N        DC    F'10'                     NB DE MOTS DU TABLEAU
000080 0000000FFFFFFFFD                 64 TAB      DC    F'15,-3,2,7,11,-1,0,9,2,10'
000088 0000000200000007
000090 0000000BFFFFFFFF
000098 0000000000000009
0000A0 000000020000000A
0000A8                                  65 SAVE     DS    18F                       SAVE AREA
0000F0                                  66 SNAPFIN  DS    0H
                                        67 POUB     DCB   DSORG=PS,RECFM=VBA,MACRF=(W),BLKSIZE=882,
                                                          LRECL=125,DDNAME=POUBELLE
000000                                 118          END   DEBUT

PSW AT ENTRY TO SNAP   078D1000 00087076      ILC 2    INTC 0033

REGS AT ENTRY TO SNAP

     FLTR 0-6     0000000000000000      0000000000000000      0000000000000000      0000000000000000

     REGS 0-7     00000034  A0087050   00000004  00000024     00000028  FFFFFFFF   00084F88  000000FF
     REGS 8-15    00000000  00084EB0   00084FD0  00087000     00087010  00087088   0000C758  00000000

-STORAGE
087000                                           90ECD00C 18CF50D0 C0AC182D 41D0C0A8   *                ................*
087020     50020008 4510C01C 8F087100 0A134140   00004120 00048830 C07C0630 89300002   *.K.............................*
087040     41000000 5A04C080 8742C034 4510C060   00720000 0000A400 00087100 00000000   *................................*
087060     00087068 00000000 00087010 80087100   9680101C 0A330700 4510C070 80087100   *................................*
087080     0A145800 C0AC98EC D00C07FE 0000000A   0000000F FFFFFFFD 00000002 00000007   *................................*
0870A0     0000000B FFFFFFFF 00000000 00000009   00000002 0000000A 00000000 00087F88   *................................*
0870C0     00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
       LINE 0870E0 SAME AS ABOVE
087100     00000000                                                                    *.....                           *
```

تمرين 4.13 ـ كما في التمرين 6.11 ، نستعمل كون العمليات مع التعليمات MVC، CLC . . . تجري بايتة بعد بايتة مع انتشار من اليسار إلى اليمين .

```
CLI     ZONE,X'00'          ( مقارنة أوّل بايتة مع 0 )
BNE     NONZERO
CLC     ZONE+1(L'ZONE-1),ZONE
BNE     NONZERO
BE      ZERO
```

التعرّف إلى « الفراغ blank) يتمّ عبر المقارنة مع X'40' .

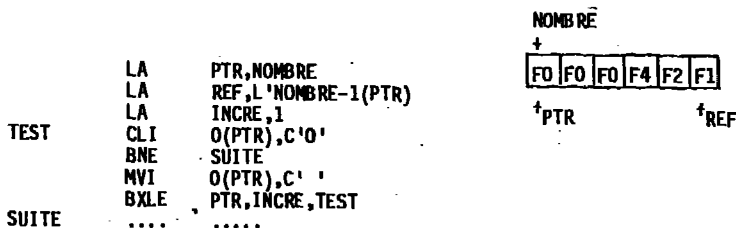
تمرين 5.13 ـ

```
XI      *+5,X'F0'
NOP     ETIQ
```

تمرين 6.13 ـ

```
NOP     ETIQ
XI      *-3,X'F0'
```

تمرين 7.13 . INCRE وREF يشكّلان مرصفاً مزدوجاً يحتوي الزيادة.والمرجع بالنسبة للتعليمة BXLE وNOMBRE هو عنوان العدد. أمّا PTR فهو مرصف مصوّب (مؤشّر) .



```
        LA      PTR,NOMBRE
        LA      REF,L'NOMBRE-1(PTR)
        LA      INCRE,1
TEST    CLI     0(PTR),C'0'
        BNE     SUITE
        MVI     0(PTR),C' '
        BXLE    PTR,INCRE,TEST
SUITE   ....    .....
```


تمرين 8.13 .

```
INDIC    DC     X'00'
INDLEC   EQU    X'80'
INDECR   EQU    X'40'
INDWAIT  EQU    X'20'
         ---    ----
         OI     INDIC,INDWAIT
         OI     INDIC,INDLEC+INDWAIT
         NI     INDIC,X'FF'-(INDLEC+INDECR)
         --     ----
         TM     INDIC,INDWAIT
         BO     ALPHA
         --     -----
         TM     INDIC,INDLEC+INDWAIT
         BO     BETA
         BM     GAMMA
         BZ     DELTA
```

مع هذا الحلّ فإنّ التعديل المتعلّق بـ INDLEC يُترجَم بواسطة :

```
INDLEC      EQU      X'01'
```

لا تتأثّر أي تعليمة تحديد موضوع أو اختيار . والأمر لا يكون كذلك إن نحن لم نستعمل EQU لتحديد المؤشّرات الثنائية ، فحينئذٍ لكان الكود مجمّداً بسبب ظهور القيم X'80' . . . في قلب التعليمات نفسها .

من جهة أخرى فإنّ هذه التقنية تحوّل التعليمات لأن تصبح موثّقة ذاتياً .

تمرين 1.14 ـ

```
SLL    R,32
SRL    R,32
```
أو

تمرين 2.14 ـ

يكتب الضرب بـ 2^3 : SLA R,3

القسمة على 16 : SRA R,4

ترافق القسمة عملية بتر (قطع) . والتمثيل بالمكمّل إلى 2 يجعل 2 / 15 + تعطي 7 في حين أنّ 2 / 15 ـ تعطي 8− .

تمرين 3.14 ـ

```
SLDA    R,0
ZERO
```
R هو مرصف مزدوج

تمرين 4.14 . أثناء عملية إزاحة دائرية إلى اليسار نحاول إعادة إدخال كلّ بتة خارجة في جهة اليمين . العمل يتمّ على مرصف مزدوج . بعد تصفير مرصف اليسار نجري إزاحة مزدوجة بشكل يسمح بأن نجد من جديد في مرصف اليسار البتات المفقودة في مرصف اليمين . وتتيح لنا تعليمة أو (OR) بإعادة وضعها في مرصف اليمين . هنا نجري إزاحة دائرية من أربعة مواقع على المرصف 7 .

```
SLL     6,32     تصفير
SLDL    6,4
OR      7,6
```

تمرين 1.18 ـ

نستعمل تعليمة TR « بالمقلوب » .

```
         TR      CLE,ARTICLE
         ---     -----
ARTICLE  DC      CL10'ABCDEFGHIJ'     ( فقرة )
CLE      DS      0CL5                 ( مفتاح )
         DC      HL1'5,6,7,1,2'
```

تمرين 2.18 -

نستعمل التعليمة TR

```
TR      CHAINE(8),TABLE
---     -----
DC      C'0123456789ABCDEF'      ( جدول )
DC      2F                       ( سلسلة )
```

ملحقات

جدول تكوين السمات.
جدول أبجدي للتعليمات
أوامر المؤول
مميزات الثوابت
كود حرفي (تذكيري) موسّع

جدول تكويد السمات

عشري	سادس عشري	حرفي تذكيري	سمة مطبوعة	بطاقة مثقوبة	عشري	سادس عشري	حرفي تذكيري	سمة مطبوعة	بطاقة مثقوبة
0	00			12-0-9-8-1	64	40	STH	بياض	لا تثقيب
1	01			12-9-1	65	41	LA		12-0-9-1
2	02			12-9-2	66	42	STC		12-0-9-2
3	03			12-9-3	67	43	IC		12-0-9-3
4	04	SPM		12-9-4	68	44	EX		12-0-9-4
5	05	BALR		12-9-5	69	45	BAL		12-0-9-5
6	06	BCTR		12-9-6	70	46	BCT		12-0-9-6
7	07	BCR		12-9-7	71	47	BC		12-0-9-7
8	08	SSK		12-9-8	72	48	LH		12-0-9-8
9	09	ISK		12-9-8-1	73	49	CH		12-8-1
10	0A	SVC		12-9-8-2	74	4A	AH		12-8-2
11	0B			12-9-8-3	75	4B	SH	(نقطة)	12-8-3
12	0C			12-9-8-4	76	4C	MH	<	12-8-4
13	0D			12-9-8-5	77	4D		(	12-8-5
14	0E	MVCL		12-9-8-6	78	4E	CVD	+	12-8-6
15	0F	CLCL		12-9-8-7	79	4F	CVB		12-8-7
16	10	LPR		12-11-9-8-1	80	50	ST	&	12
17	11	LNR		11-9-1	81	51			12-11-9-1
18	12	LTR		11-9-2	82	52			12-11-9-2
19	13	LCR		11-9-3	83	53			12-11-9-3
20	14	NR		11-9-4	84	54	N		12-11-9-4
21	15	CLR		11-9-5	85	55	CL		12-11-9-5
22	16	OR		11-9-6	86	56	O		12-11-9-6
23	17	XR		11-9-7	87	57	X		12-11-9-7
24	18	LR		11-9-8	88	58	L		12-11-9-8
25	19	CR		11-9-8-1	89	59	C		11-8-1
26	1A	AR		11-9-8-2	90	5A	A		11-8-2
27	1B	SR		11-9-8-3	91	5B	S	$	11-8-3
28	1C	MR		11-9-8-4	92	5C	M	*	11-8-4
29	1D	DR		11-9-8-5	93	5D	D	)	11-8-5
30	1E	ALR		11-9-8-6	94	5E	AL		11-8-6
31	1F	SLR		11-9-8-7	95	5F	SL		11-8-7
32	20	LPDR		11-0-9-8-1	96	60	STD	–	11
33	21	LNDR		0-9-1	97	61		/	0-1
34	22	LTDR		0-9-2	98	62			11-0-9-2
35	23	LCDR		0-9-3	99	63			11-0-9-3
36	24	HDR		0-9-4	100	64			11-0-9-4
37	25	LRDR		0-9-5	101	65			11-0-9-5
38	26	MXR		0-9-6	102	66			11-0-9-6
39	27	MXDR		0-9-7	103	67	MXD		11-0-9-7
40	28	LDR		0-9-8	104	68	LD		11-0-9-8
41	29	CDR		0-9-8-1	105	69	CD		0-8-1
42	2A	ADR		0-9-8-2	106	6A	AD		12-11
43	2B	SDR		0-9-8-3	107	6B	SD		0-8-3
44	2C	MDR		0-9-8-4	108	6C	MD	%	0-8-4
45	2D	DDR		0-9-8-5	109	6D	DD	_	0-8-5
46	2E	AWR		0-9-8-6	110	6E	AW	>	0-8-6
47	2F	SWR		0-9-8-7	111	6F	SW	?	0-8-7
48	30	LPER		12-11-0-9-8-1	112	70	STE		12-11-0
49	31	LNER		9-1	113	71			12-11-0-9-1
50	32	LTER		9-2	114	72			12-11-0-9-2
51	33	LCER		9-3	115	73			12-11-0-9-3
52	34	HER		9-4	116	74			12-11-0-9-4
53	35	LRER		9-5	117	75			12-11-0-9-5
54	36	AXR		9-6	118	76			12-11-0-9-6
55	37	SXR		9-7	119	77			12-11-0-9-7
56	38	LER		9-8	120	78	LE		12-11-0-9-8
57	39	CER		9-8-1	121	79	CE		8-1
58	3A	AER		9-8-2	122	7A	AE		8-2
59	3B	SER		9-8-3	123	7B	SE	#	8-3
60	3C	MER		9-8-4	124	7C	ME		8-4
61	3D	DER		9-8-5	125	7D	DE		8-5
62	3E	AUR		9-8-6	126	7E	AU	=	8-6
63	3F	SUR		9-8-7	127	7F	SU	"	8-7

جدول تكويد السمات

عشري	سادس عشري	حرفي تذكيري	سمة مطبوعة	بطاقة	عشري	سادس عشري	حرفي تذكيري	سمة مطبوعة	بطاقة مثقوبة
128	80	SSM		12-0-8-1	192	C0			12-0
129	81			12-0-1	193	C1		A	12-1
130	82	LPSW		12-0-2	194	C2		B	12-2
131	83			12-0-3	195	C3		C	12-3
132	84	WRD		12-0-4	196	C4		D	12-4
133	85	RDD		12-0-5	197	C5		E	12-5
134	86	BXH		12-0-6	198	C6		F	12-6
135	87	BXLE		12-0-7	199	C7		G	12-7
136	88	SRL		12-0-8	200	C8		H	12-8
137	89	SLL		12-0-9	201	C9		I	12-9
138	8A	SRA		12-0-8-2	202	CA			12-0-9-8-2
139	8B	SLA		12-0-8-3	203	CB			12-0-9-8-3
140	8C	SRDL		12-0-8-4	204	CC			12-0-9-8-4
141	8D	SLDL		12-0-8-5	205	CD			12-0-9-8-5
142	8E	SRDA		12-0-8-6	206	CE			12-0-9-8-6
143	8F	SLDA		12-0-8-7	207	CF			12-0-9-8-7
144	90	STM		12-11-8-1	208	D0			11-0
145	91	TM		12-11-1	209	D1	MVN	J	11-1
146	92	MVI		12-11-2	210	D2	MVC	K	11-2
147	93	TS		12-11-3	211	D3	MVZ	L	11-3
148	94	NI		12-11-4	212	D4	NC	M	11-4
149	95	CLI		12-11-5	213	D5	CLC	N	11-5
150	96	OI		12-11-6	214	D6	OC	O	11-6
151	97	XI		12-11-7	215	D7	XC	P	11-7
152	98	LM		12-11-8	216	D8		Q	11-8
153	99			12-11-9	217	D9		R	11-9
154	9A			12-11-8-2	218	DA			12-11-9-8-2
155	9B			12-11-8-3	219	DB			12-11-9-8-3
156	9C	SIO		12-11-8-4	220	DC	TR		12-11-9-8-4
157	9D	TIO		12-11-8-5	221	DD	TRT		12-11-9-8-5
158	9E	HIO		12-11-8-6	222	DE	ED		12-11-9-8-6
159	9F	TCH		12-11-8-7	223	DF	EDMK		12-11-9-8-7
160	A0			11-0-8-1	224	E0			0-8-2
161	A1			11-0-1	225	E1			11-0-9-1
162	A2			11-0-2	226	E2		S	0-2
163	A3			11-0-3	227	E3		T	0-3
164	A4			11-0-4	228	E4		U	0-4
165	A5			11-0-5	229	E5		V	0-5
166	A6			11-0-6	230	E6		W	0-6
167	A7			11-0-7	231	E7		X	0-7
168	A8			11-0-8	232	E8		Y	0-8
169	A9			11-0-9	233	E9		Z	0-9
170	AA			11-0-8-2	234	EA			11-0-9-8-2
171	AB			11-0-8-3	235	EB			11-0-9-8-3
172	AC	STNSM		11-0-8-4	236	EC			11-0-9-8-4
173	AD	STOSM		11-0-8-5	237	ED			11-0-9-8-5
174	AE	SIGP		11-0-8-6	238	EE			11-0-9-8-6
175	AF	MC		11-0-8-7	239	EF			11-0-9-8-7
176	B0			12-11-0-8-1	240	F0	SRP	0	0
177	B1	LRA		12-11-0-1	241	F1	MVO	1	1
178	B2			12-11-0-2	242	F2	PACK	2	2
179	B3			12-11-0-3	243	F3	UNPK	3	3
180	B4			12-11-0-4	244	F4		4	4
181	B5			12-11-0-5	245	F5		5	5
182	B6	STCTL		12-11-0-6	246	F6		6	6
183	B7	LCTL		12-11-0-7	247	F7		7	7
184	B8			12-11-0-8	248	F8	ZAP	8	8
185	B9			12-11-0-9	249	F9	CP	9	9
186	BA	CS		12-11-0-8-2	250	FA	AP		12-11-0-9-8-2
187	BB	CDS		12-11-0-8-3	251	FB	SP		12-11-0-9-8-3
188	BC			12-11-0-8-4	252	FC	MP		12-11-0-9-8-4
189	BD	CLM		12-11-0-8-5	253	FD	DP		12-11-0-9-8-5
190	BE	STCM		12-11-0-8-6	254	FE			12-11-0-9-8-6
191	BF	ICM		12-11-0-8-7	255	FF			12-11-0-9-8-7

جدول أبجدي للتعليمات

النسق	منطقة العوامل	Format	منطقة العوامل
RR	R_1,R_2	SI	$D_1(B_1),I_2$
RR-M	M_1,R_2	S	$D_2(B_2)$
RR-1	R_1	SS-1	$D_1(L,B_1),D_2(B_2)$
RR-I	I	SS-2	$D_1(L_1,B_1),D_2(L_2,B_2)$
RX	$R_1,D_2(X_2,B_2)$	SS-3	$D_1(L_1,B_1),D_2(B_2),I_3$
RX-M	$M_1,D_2(X_2,B_2)$		
RS	$R_1,R_3,D_2(B_2)$		
RS-M	$R_1,M_3,D_2(B_2)$		

R, X مراصف
D إزاحة
M قناع بأربعة بتات
I قيمة فورية
L طول

الدّالة (الوظيفة)	حرفي تذكيري N	COP سادس عشري	النسق	يحدّد موضع CC
Add	AR	1A	RR	*
Add	A	5A	RX	*
Add Decimal	AP	FA	SS-2	*
Add Halfword	AH	4A	RX	*
Add Logical	ALR	1E	RR	*
Add Logical	AL	5E	RX	*
AND	NR	14	RR	*
AND	N	54	RX	*
AND	NI	94	SI	*
AND	NC	D4	SS-1	*
Branch and Link	BALR	05	RR	
Branch and Link	BAL	45	RX	
Branch on Condition	BCR	07	RR-M	
Branch on Condition	BC	47	RX-M	
Branch on Count	BCTR	06	RR	
Branch on Count	BCT	46	RX	
Branch on Index High	BXH	86	RS	
Branch on Index Low or Equal	BXLE	87	RS	
Compare	CR	19	RR	*
Compare	C	59	RX	*
Compare and Swap	CS	BA	RS	*
Compare Decimal	CP	F9	SS-2	*
Compare Double and Swap	CDS	BB	RS	*
Compare Halfword	CH	49	RX	*
Compare Logical	CLR	15	RR	*
Compare Logical	CL	55	RX	*
Compare Logical	CLC	D5	SS-1	*
Compare Logical	CLI	95	SI	*
Compare Logical Characters under Mask	CLM	BD	RS-M	*
Compare Logical Long	CLCL	0F	RR	*
Convert to Binary	CVB	4F	RX	
Convert to Decimal	CVD	4E	RX	

الدّالة (الوظيفة) Fonction	حرفي تذكيري Mnémo-nique	سادس عشري hexa-decimal	النسق Format	محدّد موضع CC
Divide	DR	1D	RR	
Divide	D	5D	RX	
Divide Decimal	DP	FD	SS-2	
Edit	ED	DE	SS-1	*
Edit and Mark	EDMK	DF	SS-1	*
Exclusive OR	XR	17	RR	*
Exclusive OR	X	57	RX	*
Exclusive OR	XI	97	SI	*
Exclusive OR	XC	D7	SS-1	*
Execute	EX	44	RX	
Insert Character	IC	43	RX	
Insert Characters under Mask	ICM	BF	RS-M	*
Load	LR	18	RR	
Load	L	58	RX	
Load Address	LA	41	RX	
Load and Test	LTR	12	RR	*
Load Complement	LCR	13	RR	*
Load Halfword	LH	48	RX	
Load Multiple	LM	98	RS	
Load Negative	LNR	11	RR	*
Load Positive	LPR	10	RR	*
Monitor Call	MC	AF	SI	
Move	MVI	92	SI	
Move	MVC	D2	SS-1	
Move Long	MVCL	0E	RR	*
Move Numerics	MVN	D1	SS-1	
Move with Offset	MVO	F1	SS-2	
Move Zones	MVZ	D3	SS-1	
Multiply	MR	1C	RR	
Multiply	M	5C	RX	
Multiply Decimal	MP	FC	SS-2	
Multiply Halfword	MH	4C	RX	
OR	OR	16	RR	*
OR	O	56	RX	*
OR	OI	96	SI	*
OR	OC	D6	SS-1	*
Pack	PACK	F2	SS-2	
Set Program Mask	SPM	04	RR-1	
Shift and Round Decimal	SRP	F0	SS-3	*
Shift Left Double	SLDA	8F	RS	*
Shift Left Double Logical	SLDL	8D	RS	
Shift Left Single	SLA	8B	RS	*
Shift Left Single Logical	SLL	89	RS	
Shift Right Double	SRDA	8E	RS	*
Shift Right Double Logical	SRDL	8C	RS	
Shift Right Single	SRA	8A	RS	*
Shift Right Single Logical	SRL	88	RS	
Store	ST	50	RX	
Store Character	STC	42	RX	
Store Characters under Mask	STCM	BE	RS-M	
Store Clock	STCK	B205	S	*
Store Halfword	STH	40	RX	
Store Multiple	STM	90	RS	

Subtract	SR	1B	RR	*
Subtract	S	5B	RX	*
Subtract Decimal	SP	FB	SS-2	*
Subtract Halfword	SH	4B	RX	*
Subtract Logical	SLR	1F	RR	*
Subtract Logical	SL	5F	RX	*
Supervisor Call	SVC	0A	RR-I	
Test and Set	TS	93	S	*
Test under Mask	TM	91	SI	*
Translate	TR	DC	SS-1	
Translate and Test	TRT	DD	SS-1	*
Unpack	UNPK	F3	SS-2	
Zero and Add Decimal	ZAP	F8	SS-2	*

تعليمات حسابية بالفاصلة المتحركة

Add Normalized, Extended	AXR	36	RR	*
Add Normalized, Long	ADR	2A	RR	*
Add Normalized, Long	AD	6A	RX	*
Add Normalized, Short	AER	3A	RR	*
Add Normalized, Short	AE	7A	RX	*
Add Unnormalized, Long	AWR	2E	RR	*
Add Unnormalized, Long	AW	6E	RX	*
Add Unnormalized, Short	AUR	3E	RR	*
Add Unnormalized, Short	AU	7E	RX	*
Compare, Long	CDR	29	RR	*
Compare, Long	CD	69	RX	*
Compare, Short	CER	39	RR	*
Compare, Short	CE	79	RX	*
Divide, Long	DDR	2D	RR	
Divide, Long	DD	6D	RX	
Divide, Short	DER	3D	RR	
Divide, Short	DE	7D	RX	
Halve, Long	HDR	24	RR	
Halve, Short	HER	34	RR	
Load and Test, Long	LTDR	22	RR	*
Load and Test, Short	LTER	32	RR	*
Load Complement, Long	LCDR	23	RR	*
Load Complement, Short	LCER	33	RR	*
Load, Long	LDR	28	RR	
Load, Long	LD	68	RX	
Load Negative, Long	LNDR	21	RR	*
Load Negative, Short	LNER	31	RR	*
Load Positive, Long	LPDR	20	RR	*
Load Positive, Short	LPER	30	RR	*
Load Rounded, Extended Long	LRDR	25	RR	
Load Rounded, Long to Short	LRER	35	RR	
Load, Short	LER	38	RR	
Load, Short	LE	78	RX	
Multiply, Extended	MXR	26	RR	
Multiply, Long	MDR	2C	RR	
Multiply, Long	MD	6C	RX	
Multiply, Long/Extended	MXDR	27	RR	
Multiply, Long/Extended	MXD	67	RX	
Multiply, Short	MER	3C	RR	
Multiply, Short	ME	7C	RX	
Store, Long	STD	60	RX	

Store, Short	STE	70	RX	
Subtract Normalized, Extended	SXR	37	RR	*
Subtract Normalized, Long	SDR	2B	RR	*
Subtract Normalized, Long	SD	6B	RX	*
Subtract Normalized, Short	SER	3B	RR	*
Subtract Normalized, Short	SE	7B	RX	*
Subtract Unnormalized, Long	SWR	2F	RR	*
Subtract Unnormalized, Long	SW	6F	RX	*
Subtract Unnormalized, Short	SUR	3F	RR	*
Subtract Unnormalized, Short	SU	7F	RX	*

أوامر المؤول

تعريف المعطيات	DC DS CCW	Assemblage conditionnel تأويل مشروط	MACRO MNOTE MEXIT MEND
تقطيع	START CSECT DSECT COM ENTRY EXTRN		ACTR AGO AIF ANOP GBLA GBLB GBLC
تعريف المراصف القاعدية	USING DROP		LCLA LCLB LCLC
مراقبة اللائحة	TITLE EJECT SPACE PRINT		SETA SETB SETC
مراقبة البرنامج	EQU ORG LTORG CNOP END COPY PUNCH REPRO ISEQ ICTL PUSH POP OPSYN		

مميزات الثوابت

بتر أو ملء إلى	يتميّز بـ	حدّ الإصطفاف	الطول الضمني	النوع
اليمين	سمات	بايتة	-	C
اليسار	أرقام سادس عشرية	بايتة	-	X
اليسار	أرقام ثنائية	بايتة	-	B
اليسار	أرقام عشرية	كلمة	4	F
اليسار	أرقام عشرية	نصف كلمة	2	H
اليمين	أرقام عشرية	كلمة	4	E
اليمين	أرقام عشرية	كلمة مزدوجة	8	D
اليمين	أرقام عشرية	كلمة مزدوجة	16	L
اليسار	أرقام عشرية	بايتة	-	P
اليسار	أرقام عشرية	بايتة	-	Z
اليسار	تعبير	كلمة	4	A
اليسار	تعبير	نصف كلمة	2	Y
-	تعبير	نصف كلمة	2	S
اليسار	مز قابل للنقل	كلمة	4	V

الكود الحرفي موسع

القناع	التعليمة المولّدة	المعنى . . .	كود العملية الحرفي
1111	BC 15,... BCR 15,...	تفريع غير مشروط	B ... BR ...
0000	BC 0,... BCR 0,...	لا عملية	NOP ... NOPR ...
		. . . بعد تعليمات المقارنة تفريع إذا كان :	
0010	BC 2,... BCR 2,...	(*) 2 المتأثر 2 > المتأثر 1 (*) Op	BH ... BHR ...
0100	BC 4,... BCR 4,...	" < "	BL ... BLR ...
1000	BC 8,... BCR 8,...	" = "	BE ... BER ...
1101	BC 13,... BCR 13,...	" ≤ "	BNH ... BNHR ...
1011	BC 11,... BCR 11,...	" ≥ "	BNL ... BNLR ...
0111	BC 7,... BCR 7,...	" ≠ "	BNE ... BNER ...
		. . . بعد التعليمات الحسابية تفريع إذا كانت النتيجة	
0001	BC 1,... BCR 1,...	فيض عن السعة . . .	BO ... BOR ...
0010	BC 2,... BCR 2,...	... > 0	BP ... BPR ...
0100	BC 4,... BCR 4,...	... < 0	BM ... BMR ...
1101	BC 13,... BCR 13,...	... ≤ 0	BNP ... BNPR ...
1011	BC 11,... BCR 11,...	... ≥ 0	BNM ... BNMR ...
0111	BC 7,... BCR 7,...	... ≠ 0	BNZ ... BNZR ...
1000	BC 8,... BCR 8,...	... = 0	BZ ... BZR ...

(*) المقصود هما المتأثران 1 و2 في تعليمة المقارنة .

ملاحظة : الكود الحرفي التذكيري المنتهي بحرف R يولّد تعليمات من النسق RR . المرصف المذكور يحتوي على عنوان التفريع .

مثلاً : BR 3 تفريع غير مشروط إلى العنوان الواقع في المرصف 3 .

B ALPHA تفريع غير مشروط إلى العنوان ALPHA .

ترجمة الملاحظيات الواردة

في بعض البرامج الموجودة في الكتاب

السطر	الملاحظية	الصفحة
5	ثوابت سمات . لا يوجد اصطفاف خاص . الطول 256	69
6	تأطير إلى اليسار . بتر إلى اليمين	
8	بتر إلى اليمين .	
9	تأطير إلى اليسار تكمله فراغات .	
10	توليد فاصلة عليا واحدة .	
11	نفس الملاحظة	
12	تكرار وبتر	
15	ثوابت سادس عشرية . تأطير إلى اليمين . بتر إلى اليسار .	
16	طول ضمني	
17	طول ظاهر .	
18	بتر .	
21	ثوابت ثنائية . الطول الأقصى 256 بايتة تأطير إلى اليمين .	
22	تكمله أصفار إلى اليسار . اصطفاف على البايتة .	
23	ثنائي	
24	بتر إلى اليسار .	
25	بتر .	
26	تكرار .	
29	ثوابت بالفاصلة الثابتة على كلمة (F) أو نصف كلمة (H) .	
30	اصطفاف على الكلمة أو نصف الكلمة . عندما يكون الطول	
31	محدّدا لا يعود هناك اصطفاف . الثابتة هي بالنظام العشري	
34	إزاحة 3 بتات إلى اليسار (8 *)	
36	إزاحة 3 بتات إلى اليمين (8/)	
39	مدور أعلى	
40	مدور أصغر .	
42	تعديل الطول LONG والاصطفاف ALIGN .	
43	إزاحة بتتين إلى اليسار .	
49	ثوابت بالفاصلة المتحركة وبالدقة البسيطة . اصطفاف على الكلمة	70
50	تأطير إلى اليمين . لا بتر . القيمة مدوّرة .	
51	الطول الضمني 4 بايتات .	

السطر	الملاحظة	الصفحة
52	بفاصلة متحرّكة	70
57	ثوابت بالفاصلة المتحرّكة وبالدقّة المزدوجة	
58	اصطفاف على الكلمة المزدوجة . تأطير إلى اليمين . لا بتر	
59	القيمة مدوّرة . الطول الضمني 8 بايتات	
66	ثوابت بالفاصلة المتحرّكة وبالدقّة الرباعية	
67	اصطفاف على الكلمة المزدوجة . الطول الضمني 16 بايتة .	
68	لا بتر . القيمة مدوّرة . أسّ من 85− إلى 75+ .	
73	ثوابت عشرية . الطول الأقصى يبلغ	
74	16 بايتة . الإشارة تقع في الربع الأيسر	
75	من البايتة اليمنى الأخيرة . تأطير إلى اليمين . بتر إلى اليسار .	
76	X'F أو X'C' في موقع الإشارة يُعتبران مثل +	
77	X'D أو X'E' في موقع الإشارة يُعتبران مثل −	
78	لا يتم ترجمة الفاصلة العشرية أبداً إلى الثنائي .	
79	تأطير إلى اليمين . بتر إلى اليسار .	
80	الطول الضمني .	
82	بتر إلى اليسار	
86	الثوابت العشرية المكثّفة (Packed)	
87	نفس قواعد الثوابت السابقة .	
88	تقع الإشارة في الربع الأيمن الأخير .	
2	رمز خارجي	73
6	رمز قابل للنقل	
9	ثوابت عنوان من النوع A	
10	تُكتب DC A (تعبير مطلق أو قابل للنقل)	
11	اصطفاف على الكلمة . الطول الضمني 4 بايتات .	
12	الأطوال الظاهرة الممكنة هي من 1 إلى 4 بايتات .	
13	بتر إلى اليسار . ممكن التحديد في كود حرفي .	
18	طول ظاهر	
20	رمز خارجي	
23	ثوابت عنوان من النوع Y	
24	تُكتب DC Y (تعبير مطلق أو قابل للنقل)	
25	اصطفاف على نصف الكلمة . الطول الضمني نصف كلمة .	
26	الأطوال الظاهرة الممكنة هي من 1 أو 2 بايتة .	
27	بتر إلى اليسار . ممكن التحديد في كود حرفي .	
29	لاحظوا أنّ النجمتين	
30	تساويانْ B وB + 2	
31	الطول الظاهر	
32	بتر إلى اليسار	
35	ثوابت عنوان من النوع S	
36	تُكتب DC S (تعبير مطلق) .	
37	أو DC S (تعبير قابل للنقل)	
38	أو DC S (تعبير مطلق (تعبير مطلق)) .	

السطر	الملاحظة	الصفحة
39	مؤوّلة في نصف كلمة . مصطفّة على نصف الكلمة .	73
40	لا يمكن تحديدها في كود حرفي .	
42	القاعدة (Base) 0 ، الإزاحة (Déplacement) = 1024	
43	قاعدة وإزاحة RELOC	
49	ثوابت عنوان من النوع V	
50	تُستعمل فقط للعناوين الخارجية من النوع اسم البرنامج NOU-DE-PROG .	
51	تكتب V DC (رمز خارجي قابل للنقل)	
52	لا يرد الرمز القابل للنقل في أمر خارجي .	
53	الطول الضمني 4 بايتات . معدّل الطول = 3 أو 4 .	
54	اصطفاف على حدّ كلمة ، بإمكانه أن يظهر في كود حرفي .	
55	يولّد المؤوّل كلمة صفر .	
3	متالية الدخول	79
4 و5	حفظ المراصف من شحن مرصف القاعدة	
6	R12 = مرصف القاعدة	
7	البرنامج المنادي	
14	اصطفاف كلمة	
18	(1) القاعدة 12 ظاهرة	
19	(2) القاعدة 12 ظاهرة	
22, 21 و23	كلّ التعليمات من (3) حتّى (7) تشحن X'89ABCDEF' في المرصف 3 . الكتابة (3) هي الوحيدة المستقلّة عن مكان ALPHA بالنسبة إلى عنوان القاعدة .	
24 و25	(3) استعمال تعبير قابل للنقل . قاعدة ضمنية .	
26	(4) تعليمة تماثل رمزاً مطلقاً .	
30	(7) استعمال كود حرفي	
32	(8) "8" هي عبارة عن إزاحة	
24	(9) خطأ اصطفاف	
36	(10) "12" هي عبارة عن إزاحة	
37	(11) "12" هي عبارة عن مرصف قاعدي	
38	(12) خطأ في النحو	
39	(13) خطأ في النحو	
40	(14) خطأ في النحو	
41	(15) 12 هي عبارة عن مؤشّر	
3	مؤشّر (مصوب) إلى عنصر من TAB	111
4	مرصف إضافة لـ BXLE	
5	مرصف مرجع لـ BXLE	
6	مرصف عمل	
11	القاعدة = المرصف 12	
16	تصفير (إعداد)	
23	طول الكلمة	

السطر	الملاحظة	الصفحة
		111
47	حلقة مسح الجدول	
48	و49 في حال عدم التبديل يتم فرز (ترتيب) TAB	
52	تصفير المؤشر	
53	تصفير مرجع BXLE	
56	عنصر ايسر في مرصف العمل	
57	مقارنة	
60	تبديل	
62	تحديد موقع INDIC	
101	منطقة المعطيات	
102	عدد عناصر TAB	
105	إعداد INDIC	
39	مؤشّر بداية الجدول الثانوي	115
40	مؤشّر نهاية الجدول الثانوي	
41	مؤشّر المتصف والرتبة	
42	مرصف العمل	
43	طول العنصر	
62	عدد عمليات التكرار في البرنامج	
76	إعداد	
80	حساب عنوان العنصر الوسط (المتصف)	
84	قسمة على L * 2	
85	(PTRELEM) = عدد العناصر في الجدول الثانوي	
87	إذا 0 نرغم حتّى 1	
89	ضرب بـ >	
91	مقارنة	
92	تفريع إذا كان ELEM < (MOT)،	
93	تفريع إذا كان ELEM > (MOT)	
95	وجدنا العنصر حساب رتبة العنصر = (MOT)	
98	قسمة على الطول	
100	طباعة الرتبة والقيمة	
129	لم نجده	116
153	منطقة المعطيات	
154	عدد كلمات الجدول	
155	طول العنصر	
3	حفظ مراصف المنادي	160
5	تعريف وشحن مرصف القاعدة	
6	نأخذ المرصف 12 كقاعدة	
7	عنوان PROGJ في 12	
8	حفظ R13 في المنطقة SAVE AREA من البرنامج .	

الصفحة

10 و11 حفظ عنوان المنطقة SAVE AREA من هذا البرنامج في المنطقة AREA من المنادي 160
16 تعريف المنطقة SAVE AREA
22 متتالية نداء PROGK
29 متتالية العودة إلى PROGI

فهرست

الموضوع **الصفحة**

هذا الكتاب

تعتبر لغة المؤوّل (الأسمبلر) من العناصر الأساسية في التفكير حول طريقة البرمجة بإحدى اللغات المتطوّرة فهي تتيح لنا فهماً مفصّلاً لأواليات الحاسب وليس بالإمكان الاستغناء عنها في إعداد المعلوماتي .

وتتجلّى ضرورة إستعمال لغة المؤوّل ، بالرغم من قوّة اللغات المتطوّرة ، عندما يوجد إلزامات بالنسبة لفترات الإجابة (بعض البرامج الكبيرة ، أنظمة التشغيل ، المصرّفات ، الوقت الحقيقي ، . . .) أو بالنسبة لحجم الذاكرة (الحاسبات الصغيرة والمتوسّطة) ، أو أيضاً إلزامات تعود إلى عدم كفاية إمكانيات البرامج (فورتران ، باسيك) .

من جهة أخرى ، سوف يجد مستعملو الميكرومعلوماتية في تطبيق لغة المؤوّل حلاً ممتازاً لما يعترضهم من مشاكل .

يتوجّه هذا الكتاب إلى الطلاّب والممارسين الذين يرغبون بتعميق معرفتهم في مجال المعلوماتية . وهو يتكوّن من فصول قصيرة ويبتدىء انطلاقاً من ملاحظات بسيطة جدّاً على حاسبة الجيب ، بشكل يقود معه القارىء شيئاً فشيئاً ، لا سيّما بفضل التمارين المحلولة والمفاهيم الأساسية في بنية الآلة ، إلى دراسة المؤوّل والماكرو ـ لغة . ولا شكّ أنّه بالإمكان استعماله كمرجع ولتدريس متعلّق بسلسلة الآلات المعتمدة كأمثلة (سلاسل 4000 ، 3000 ، IBM370) ولكنّه وضع كي يكون دليلاً عاماً يوجّه بطريقة سليمة أيّ برمجة بلغة المؤوّل .

المؤسسة الجامعية للدراسات والنشر والتوزيع